Computer algebra approaches to enzyme kinetics

Bayram, Mustafa

Award date:
1993

Awarding institution:
University of Bath

COMPUTER ALGEBRA APPROACHES TO ENZYME KINETICS

submitted by

Mustafa Bayram

for the degree of Ph.D

of the

University of Bath

1993

Signature of Author ..... M. A. Bayram .....

Mustafa Bayram

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

Abstract

To date, analyses of properties of enzymes have usually been conducted on isolated enzymes in *vitro* and by performing initial rate experiments. Since biochemical properties of both the individual enzymes and the pathways in which they act may depend on interactions with other enzymes and metabolites in *vivo*, an approach whereby systems of enzymes could be studied simultaneously would be advantageous. An approach by which this could possibly be achieved, using advances in computer algebra, has been described [Bennett, Davenport and Sauro, 1988].

This thesis describes an application of such techniques to a biochemical system. Several approaches were used:

- A steady state approach to produce rate laws for complete systems of enzyme reactions. This requires use of a computer algebra system to solve the rate laws of the individual enzymes simultaneously.
- A conventional numerical simulation was performed using an interface program between REDUCE and NAG to evaluate FORTRAN forms of the enzyme reversible rate equations and their higher derivatives, which were solved numerically.
- Calculation of a Gröbner basis was used to obtain a rate law for the overall flux of the coupled systems in terms of one metabolite. Time course data was then used in simulation by this method.
- Standard fitting techniques were used to obtain estimates of kinetic parameters using the overall rate law obtained using a Gröbner basis.
- Differentiation was used to decide which kinetic parameters could be fitted to the rate law with low error. The advantages and limitations of this approach are discussed.
- Computer algebra techniques were applied to metabolic control properties of coupled systems. Flux and concentration control coefficients and enzyme elasticities were calculated symbolicly and fitted to experimental data.
- Computer algebra techniques were applied to derive rate equations for arbitrarily complex enzyme mechanisms.

The major difficulties encountered in parameter estimation were in scaling the problem so that minimisation routines were sensitive to perturbations in the initial estimates entered, and in finding the degree of tolerance required to give acceptable results.

Acknowledgements

I would like to gratefully acknowledge the help given by my supervisors Dr. J. P. Bennett and Dr. G. C. Smith. This work would not have been possible without their help. I thank Dr. M. C. Dewar and Dr. J. A. Abbott for their advice and assistance regarding the computer algebra system and numerical analysis package (NAG). I also thank Dr D. A. Fell for his discussion on my project.

I would like to thank to School of Mathematical Sciences for facilitating my work.

I also would like to thank D.L. Fisher who ptovided all the experimental data for this project.

I would like to express my thanks to University of Atatürk and the government of Turkey for their financial support.

Finally, my thanks go to my family for their patience and sincere encouragement.

Contents

Glossary

Allosteric enzyme: some enzymes are modulated by the action of effectors that bind at sites distinct from the active site. This allows the effectors to act from metabolically distant pathways. The enzymes that are regulated in this manner are termed *"allosteric enzymes"*.

Binary complex: complex formed by association of two species.

Catalytic constant: limiting rate divided by total enzyme concentration.

Inhibition: tendency of an enzyme-catalysed reaction to proceed more slowly in the presence of a species called an inhibitor.

Inhibition constant: it represents the metabolite's tendency to bind the enzyme at the wrong stage.

Limiting rate: rate approached by an enzyme-catalysed reaction when the substrate concentrations become very large.

Maximum velocity: obsolete (and misleading) term for limiting rate.

Michaelis constant: substrate concentration at which the rate of an enzyme-catalysed reaction is half of the limiting rate.

Molarity: the molarity of a solution is defined as the number of moles of solute dissolved in 1 liter of solution (molarity = moles(n)/liters(V)).

Rate constant: quantity that when multiplied by the appropriate concentration or concentrations gives the rate of an elementary step of a reaction (or, loosely, the rate of

a composite reaction treated as if it were an elementary step).

Steady state: state in which the concentration of each intermediate in a reaction remains constant because it is removed as fast as it is produced.

Ternary complex: complex formed by association of three species.

Abbreviations and Notations

$\mathbf{A}$	:	auxiliary matrix in chapter 7, used in the determination of matrix $\mathbf{K}$. Dimensions: $m_o \times (r - mo)$
A	:	substrate in chapter 4.
$[A]$	:	concentration of A in chapter 4.
AAT	:	aspartate aminotransferase.
asp	:	aspartate.
$\mathbf{C}$	:	matrix of the normalized flux control coefficients. Dimentions: $r \times r$
C'	:	matrix of the direct flux control coefficients. Dimensions: $r \times r$.
C_{ij}	:	normalized control coefficient of the step j on the flux J_i
$C^{\mathbf{V}}_{AAT}$, $C^{\mathbf{V}}_{MDH}$	:	flux control coefficients.
C^{oaa}_{MDH}, C^{oaa}_{AAT}	:	concentration control coefficients.
E	:	enzyme.
$\mathbf{E}$	:	matrix of the normalized elasticities. Dimensions: $r \times m$.
E_o	:	total enzyme.
$[E_o]$	:	total enzyme concentration.
E'	:	matrix of the direct elasticities in chapter 7. Dimensions: $r \times m$.
glu	:	glutamate.
$\mathbf{I}$	:	identity matrix.
I_{m_o}	:	identity matrix. Dimensions: $m_o \times m_o$
J_i	:	flux i defined by $\mathbf{J_i} = v_i$ at steady state.
$\mathbf{K}$	:	matrix of the vectors of a basis of the null-space of the matrix N. Dimensions: $r \times (r - m_o)$.
K_I	:	inhibition constant.
k_i	:	rate constant.
K_M	:	Michaelis constant.
$\mathbf{L}$	:	matrix connecting $\mathbf{N_R}$ to N. Dimensions: $m \times m_o$.
L_o	:	Dimensions: $(m - m_o) \times m_o$.
m	:	number of the metabolites in the network.

m_o	:	rank of the matrix **N**.
mal	:	malate.
MDH	:	malate dehydrogenase.
N	:	stoichiometry matrix of the network. Dimensions: $m \times r$.
NAD^+	:	coenzyme form of nicotinomide adenine dinucleotide.
NADH	:	reduced form of NAD^+.
N_o	:	matrix extracted from **N**. Dimensions: $(m - m_o) \times r$.
N^*	:	matrix extracted from **N** in which the m_o first columns are independent. Dimensions: $m \times m_o$.
N_o^*	:	matrix extracted from N_o by taking same independent columns of N_o. Dimensions: $(m - m_o) \times m_o$.
N_1^*	:	residual matrix of N_o. Dimensions: $(m - m_o) \times (r - m)r$.
N_R	:	matrix extracted from **N** with m_o independent rows of **N**. Dimensions: $m_o \times r$.
N_{R_o}	:	a square invertible matrix. Dimensins: $m_o \times m_o$.
N_{R_1}	:	Dimensions: $m_o \times r - m_o$.
oaa	:	oxaloacetate.
P	:	substrate in chapter 4.
$[P]$	:	concentration of P in chapter 4.
$R^{\mathrm{V}}_{\mathrm{asp}}$	:	flux response coefficient for *asp*.
$R^{\mathrm{oaa}}_{\mathrm{asp}}$	:	concentration response coefficient for *asp*.
r	:	number of reaction in the network.
S	:	matrix of the normalized metabolite control coefficients. Dimensions: $m \times r$.
S'	:	matrix of the direct metabolite control coefficients. Dimensions: $m \times r$.
X	:	metabolite matrix. Dimensions: $m \times 1$.
X_i	:	metabolite of the network ($i = 1, \cdots, m$ in chapter 7).
$[X_o]$	:	concentration of X_o.
X_R	:	matrix extracted from **X** by taking first m_o rows. Dimensions: $m_o \times 1$
X_R'	:	residual matrix of **X**. Dimensions: $(m - m_o) \times 1$

V_{max}	:	maximum velocity.
v_i	:	rate of the step i ($i = 1, \cdots, r$, in the chapter 7).
α-kg	:	α-ketoglutarate.
ϵ_{ij}	:	normalized elasticity of the metabolite j on the rate v_i.
ϵ'_{ij}	:	direct elasticity of the metabolite j on the rate v_i.
$\epsilon^{v_1}_{oaa}$, $\epsilon^{v_2}_{oaa}$	:	elasticity coefficients.
$\epsilon^{v_1}_{ex}$	:	external elasticity coefficient for ex.
ex	:	asp, α-kg, glu, mal, NADH, NAD^+.

Chapter 1

General Introduction

1.1 Introduction

Analysis of enzyme mechanisms, although central to much of biochemistry, has always been difficult. Very often the investigation of isolated enzymes in *vitro* has been all that has been possible. To date, only control analysis [Kacser and Burns, 1973] has been able to provide any general analytical approach to dealing with systems of several enzymes.

A new approach to analysis of enzyme kinetics has recently been proposed [Bennett, Davenport and Sauro, 1988]. Measurement of enzyme kinetic parameters is usually performed by fitting experimental data to rate equations. Let us consider reversible enzyme-catalysed reactions such as

$$X_0 \overset{v}{\rightleftharpoons} X_1,$$

where right and left harpoon show reaction direction. Such reactions are characterised by a rate equation such as

$$v = \frac{\frac{V_{max}}{K_{m,f}}\left|[X_0] - \frac{[X_1]}{K_{eq}}\right|}{1 + \frac{[X_0]}{K_{m,f}} + \frac{[X_1]}{K_{m,r}}} \tag{1.1}$$

where v is the flux, i.e. the rate of conversion of X_0 to X_1 . A sequence of such reactions

can be linked together, for example

$$X_0 \overset{v_1}{\rightarrow} X_1 \overset{v_2}{\rightarrow} X_2 \overset{v_3}{\rightarrow} \cdots \overset{v_n}{\rightarrow} X_n,$$

where X_1, X_2, X_3, $\cdots$, X_{n-1} are some intermediates. It may well be hard to make accurate measurements. In this case the concentration of X_1, X_2, X_3 , $\cdots$, X_{n-1} are nevertheless assumed to be constant because the amount of concentration of intermediate is always much less than concentration of enzyme. This is called the steady-state assumption. We make our steady state assumption that $v = v_1 = v_2 = \cdots = v_n$. By solving simultaneously the equations for $v = v_1 = v_2 = \cdots = v_n$ we can eliminate the intermediate concentrations, $[X_1]$ to $[X_{n-1}]$, and obtain a formula for the flux through the entire system in terms only of substrate and product concentrations, $[X_0]$ and $[X_n]$. The individual equations for v_i are non-linear in the $[X_i]$. Our approach to the algebraic treatment of these equations is to use Buchberger's algorithm [Buchberger, 1985]. This algorithm converts a system of multivariate polynomials into a Gröbner basis, which gives direct information about the solutions. The Gröbner basis is often in triangular form, i.e, there is a polynomial in one variable, then a polynomial in first two variables, and so on. This triangular form can be exploited to find solutions by finding the roots of the last equation, substituting them into the others and so on. The resulting flux equations are large, but can be handled by a computer algebra system and then converted automatically into FORTRAN programs for specific evaluation. Derivatives of the flux equations allow us to estimate errors accurately, and identify the best range over which to carry out experimental work. For validation of these techniques, we handled three kind of enzyme kinetic models:

1. Aspartate aminotransferase (AAT) and Malate dehydrogenase (MDH) in a coupled system for which we find the flux equation is cubic in v. This used the system of figure 1-1.
2. Malate dehydrogenase (MDH) and fumarase in a coupled system (figure 1-2). The system is characterised by a quadratic flux equation.
3. Malate dehydrogenase (MDH) an isolated system which is characterised by

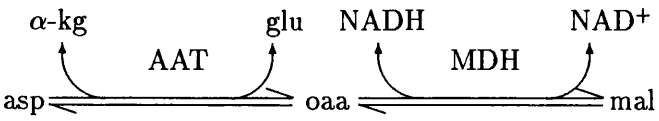


Figure 1-1: Aspartate aminotransferase (AAT) and malate dehydrogenase (MDH) in a coupled isolated system, with aspartate (asp), α-ketoglutarate (α-kg), glutamate (glu), oxaloacetate (oaa), malate (mal), NADH and NAD^+. The above notation shows that *asp* and α-kg are converted to *oaa* and *glu*, by enzymes *AAT*. Then the *oaa* and *NADH* to NAD^+ and *ma* by enzyme *MDH* in a coupled system. Right and left harpoon show the reaction can proceed each alter direction.

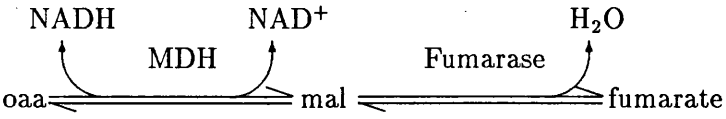


Figure 1-2: Fumarase and malate dehydrogenase (MDH) in a coupled system. In this reaction, *oaa* and *NADH* are converted to *fumarate* and NAD^+ by enzyme *MDH* and *fumarase*.

a linear flux equation (figure 1-3).

We used the computer algebra system REDUCE, running under Unix on a SUN-4/260, and numerical analysis systems (NAG).

1.2 Literature Review

When digital computers first became available, simulation became a more feasible proposition. One of the earliest group of workers in the field was associated with B. Chance, who used computers to compare biochemical simulation with experimental work from mitochandrial preparations [Chance and Williams, 1956]. Probably the most prolific worker in this field has been Garfinkel; after he had first worked with Chance, Garfinkel continued with his simulation studies, and is still active to this day. He

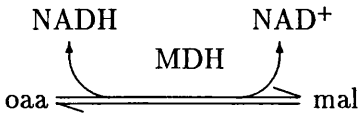


Figure 1-3: Malate dehydrogenase isolated system.

has tended to concentrate on building large models; for example, one of his models of metabolism in acidotic cardiac ischemia included 548 reactions [Achs and Garfinkel, 1982].

Garfinkel has approached simulation from two angles: he has, on the one hand, performed actual simulations of many different tissues and pathways, which has meant that he has also, in the course of his work, investigated the problems and techniques associated with simulation [Garfinkel et al, 1979]. Garfinkel has studied problems associated with the numerical analysis of metabolic models [Garfinkel and Marbach, 1977], and problems with data fitting [Waser *et al*, 1983]. he has also developed a scheme for approaching the general problem of simulating a metabolic pathway [Garfinkel et al, 1979].

Garfinkel has given a summary of computer simulation in enzyme kinetics [Garfinkel et al, 1970]. Computer simulation has also been applied to mechanistic studies of enzymes, and it can give useful indications of kinetic mechanisms. Such systems have been studied by Bates and Frieden [Bates and Frieden, 1973b]. They have discussed the following enzyme mechanism using a computer. The mechanisms involve

- a simple, single substrate enzyme,
- a single substrate, single site, allosteric enzyme,

  Some enzymes are modulated by the action of effectors that bind at sites distinct from the active site. This allows the effectors to act from metabolically distant pathways. The enzymes that are regulated in this manner are termed *"allosteric enzymes"*.

- Some two substrate enzymes,

After an enzymatic reaction has been written, it has been typed directly into the computer, and the compiler sets up the routines for the numerical integration of the differential equations describing the change, as a function of time, for each species [Bates and Frieden, 1973a-c]. Computer techniques have been applied to solve immobilized bienzymatic systems. After a set of equations has been written, then they have been solved numerically by computer [Malpiece and Sharan, 1980].

Simulation is considered to be a useful strategy in mathematical modelling of enzyme systems [Garfinkel, 1984]. Mathematical modelling of biological systems is important, because of their inherent complexity, to reduce the problem to a point where one is able to obtain useful parameters that will be predictive and aid in the design of experiments [Garfinkel, 1984; Heinrich, Rapoport, and Rapoport, 1977]. However, everything should be reduced to a level as simple as possible, but not more simple; and it has been said that

> "if mathematical models...do not consolidate knowledge, they are approximations which probably will not work."

[Albiser *et al*, 1980]. Thus complex models may be necessary. This thesis describes an attempt to analyse two enzymes simultaneously in a coupled system. But in a way that can be generalised to more complex systems.

To date, kinetic modelling has primarily been concerned with single enzyme systems [Garfinkel, 1984], often involving fitting of data to polynomial equations in enzyme modelling and in pharmacokinetics [Crabbe, 1990]. Such rational polynomial models have been analysed in some detail [Bardsley, 1977; Bardsley *et al*, 1983; Engel and Ferdinand, 1973] and have included considerations of non-linearity and curve shapes of kinetic plots of complex enzyme models. Such considerations have usually been applied only to initial-rate studies, with exceptions and have occasionally been graphically oriented [Bardsley, 1977; Crabbe, 1988] although computer modelling is more common.

Computers have been applied to regression analysis using the Michaelis-Menten initial-rate equation, to give improved fit values of parameters from estimates, generally by least-squares fitting [Crabbe, 1985] in non-linear regression, using several minimisation techniques [Bard, 1974]. Use of non-linear least squares methods for fitting equations to enzyme kinetic data has also been described [Cornish-Bowden and Endrenyi, 1981] and tested by computer simulations including considerations of inhibition, two-substrate kinetics, and pH activity profiles [Cornish-Bowden and Endrenyi, 1986]. However, the latter case was restricted to considerations of not more than four parameters and three variables that are linear in reciprocal form. Application of computers to determination of enzyme mechanism has also previously

been studied. Examples of this include the automatic generation of rate-equations [Lam and Schatz, 1978] using the King and Altman procedure [King and Altman, 1956], and computer simulation, which can give useful indications of kinetic mechanisms and kinetic parameters [Bates and Frieden, 1973a-c] and a combination of regression analysis and rate-equation computation has been applied to mechanistic studies of *E. coli* alkaline phosphatase [Waight *et al*, 1977]. Against this, however, kinetic modelling and mechanistic analysis have never been applied to more than one enzyme simultaneously using compound rate equations. The use of more powerful computer techniques has been identified as an approach to solving such complex problems [Garfinkel, 1984].

To date the only approach to analysis of multi-enzyme systems has been the descriptive though not predictive approach of control analysis [Kacser and Burns, 1973; Kacser and Porteous, 1987] and similar approaches [Crabtree and Newsholme, 1987; Savageau, 1971]. This has been applied in experimental systems [Groen *et al*, 1982; Rapoport *et al*, 1976] and this analytical approach has been considered by some to be advantageous over computer modelling [Heinrich and Rapoport, 1974]. Metabolic control analysis has been extended to matrix algebra and solution by computer [Sauro *et al*, 1987].

The present study was an analysis of the feasibility of applying recent developments in computer algebra to the kinetics of multienzyme systems, and ultimately to the analysis of such systems *in vivo*. The steady state approach, whereby reaction rates of enzymes in a linear pathway may be linked, has been identified as a candidate for computer analysis using Gröbner bases, with potential commercial ramifications [Bennett, Davenport and Sauro, 1988]. It has also been proposed that the techniques could be applied to the definition of enzyme mechanisms, where unknown, in a pathway. From analysis of the equations it would be possible to determine what measurements would need to be made to distinguish between models. A precedent for such an approach has been set [Raksanyi *et al*, 1986].

The approach is concerned with the kinetic description of metabolites (X_i, $i = 1,..,n-1$) in a system such as that shown in figure 1-4. Calculation of a Gröbner basis [Buchberger, 1985] could eliminate intermediate metabolite concentrations from

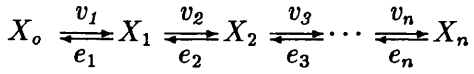


Figure 1-4: Linear n-enzyme system.

the rate-equations to obtain an equation in terms of the overall flux through the system and one or more metabolites.

The malate dehydrogenase and oxaloacetate transaminase mechanisms have been described [Fasella and Hammes, 1967; Raval and Wolfe, 1962] and rate equations describing these mechanisms in terms of the metabolite concentrations and kinetic parameters have been derived [Cornish-Bowden, 1979]. Numerical computer modelling has previously been applied to aspartate aminotransferase [Haarhoff, 1969]. This therefore seemed like a good model system to test the applicability of Gröbner bases to enzyme kinetics.

Some problems benefit from a combination of both the symbolic and numerical approach. For example it is often convenient to model a problem symbolically, deriving a family of equations which describe it, and then evaluate those equations with various numerical values. There are several surveys of such applications [Fitch, 1979; Edward,1979; Fitch, 1990], most of which are concerned with problems in physics and mechanics. Recently there has been some interest from the field of biochemistry to model enzyme reactions [Bennett and Fisher 1990; Bayram, Bennett and Dewar 1991; Bayram and Bennett 1992]. The point here is that the two approaches—symbolic and numeric—should not be viewed as divorced from one another, but rather as complementary tools for problem solving.

Chapter 2

Mathematical Techniques

It is possible to write down the equations governing a one-stage enzyme catalysed reaction quite easily, and deduce information about the steady state flux in such a system. The situation is somewhat more complicated if several such reactions form a linear chain. A particular mathematical technique of importance to this work is the calculation of Gröbner bases [Buchberger, 1985]. These are canonical representations of systems of multivariate polynomials. A consequence is that they often permit us to solve simultaneous non-linear equations. This section describes what a Gröbner basis is and how it can be used.

2.1 Gröbner Bases

We start by describing some concepts. Let K be a field; $K[x_1, \cdots, x_n]$ is a ring of n-variable polynomials over K. If $F = \{f_1, \cdots, f_n\}$ is a finite subset of $K[x_1, \cdots, x_n]$, then the ideal generated by F, Ideal(F) is

$$Ideal(F) = \left\{ \sum_{i=1}^{m} h_i f_i | h_i \in K[x_1, \cdots, x_n], f_i \in F, i = 1, \cdots, m \right\}.$$

Given a set of polynomials $F = \{f_1, f_2, \cdots, f_n\}$, we want to find the simultaneous roots. To do this we compute an equivalent set of polynomials, called a Gröbner base [Buchberger, 1985]. This set is equivalent in the sense that every root of F is a root of

the Gröbner basis. However, the Gröbner basis is typically very much easier to solve; for instance, it is often in triangular form. We start with a set of polynomials in the unknowns $x_i (i = 1, ..., n)$, which are not necessarily linear.

$$\begin{aligned} f_1(x_1, x_2, \cdots, x_n) &= 0 \\ f_2(x_1, x_2, \cdots, x_n) &= 0 \\ &\vdots \\ f_n(x_1, x_2, \cdots, x_n) &= 0 \end{aligned}$$

This is transformed into a Gröbner Basis with the form [Kalkbrener, 1987]:

$$\begin{aligned} g_1(x_1, x_2, \cdots, x_n) &= 0 \\ &\vdots \\ g_{n-1}(x_{n-1}, x_n) &= 0 \\ g_n(x_n) &= 0 \end{aligned}$$

By solving and back-substituting the equations in turn, we determine the values of x_i. This solution may be analytic, or may be numerical, since there may be more than one root of a polynomial, we may get multiple solution sets. Obviously, we can get a polynomial in x_n alone, then a polynomial in x_{n-1} and x_n alone, and so on.

Now, the question is "given polynomials sets,

$$F = \{f_1, f_2, f_3, \ldots, f_n\}$$

How can we find a Gröbner basis?" To calculate a Gröbner basis, we use a total degree ordering $(1 < x < y < x^2 < xy < y^2 < x^3 \ldots)$ or a lexicographic ordering $(1 < x < x^2 < \ldots < y < xy < \ldots < y^2 < \ldots)$.

Crucially important in constructive ideal theory [Buchberger, 1985] is the concept of reduction of a polynomial g modulo a set of polynomials F; this is our first step.

Definition: The polynomial g is reducible using f, b, u if there exists a polynomial $f \in F$ and a monomial u such that

coefficient(u $\times$ *LeadingPowerProduct(f), g)* $\neq 0$.

Then

b = *coefficient(u* $\times$ *LeadingPowerProduct(f), g)* / *LeadingCoefficient(f)*

This is shown by

$$g \xrightarrow[f,\ b,\ u]{} \cdot$$

with the notation;

Coefficient(t,g) is the coefficient of t in g,

LeadingPowerProduct(f) is the maximal power product occurring with non-zero coefficient in f.

LeadingCoefficient(f) is the coefficient of the LeadingPowerProduct(f).

Definition: The polynomial g reduces to h modulo F in one step if there exists a polynomial $f \in F$, monomial u and $b \in K$ such that

$$g \xrightarrow[f,\ b,\ u]{} h$$

and

$$h = g - b \times u \times f$$

Definition: The polynomial h is in reduced form modulo F if there is no h' such that

$$h \xrightarrow[F]{} h'$$

The polynomial h is a reduced form of g modulo F if there is a sequence of reductions

$$g = k_0 \xrightarrow[F]{} k_1 \xrightarrow[F]{} k_2 \xrightarrow[F]{} \cdots \xrightarrow[F]{} k_m = h$$

and h is in normal form modulo F.

Before we give the algorithmic application of Gröbner bases we show how it may be decided whether a given set of polynomials F defines a Gröbner basis and how Gröbner

bases may be constructed. For this the notion of an "S-polynomial" is fundamental.

Definition: Let f and g be non-zero polynomials, with leading monomials f_m and g_m. Set $h = lcm(f_m, g_m)$. The S-polynomial of f and g is

$$S(f,g) = \frac{h}{f_m} f - \frac{h}{g_m} g$$

This is just cross-multiplying to eliminate the leading monomials.

2.1.1 Buchberger's Algorithm

Buchberger's algorithm is used as a way of calculating a triangular set of polynomials. The algorithm below was devised by Buchberger, who named it the Gröbner basis algorithm.

Buchberger's Algorithm: Given a set of polynomials F, find a Gröbner basis G such that ideal(F)=ideal(G).

1. Let $G = F$, and P be the set of all unordered pairs of polynomials from G.
2. While P is non-empty do
   (a) Let (f_1, f_2) be a pair in P, and form the S-polynomial $h = S(f_1, f_2)$
   (b) Reduce $h \rightarrow h'$ with respect to G.
   (c) If $h' \neq 0$, then set $P = P \cup \{$ new pairs from $(h', G)\}$, and set $P = P - (f_1, f_2)$. Put $G = G \cup h'$.
3. Return the Gröbner basis G

All we do is search through for pairs whose S-polynomial does not reduce to zero, and if we find one, reduce it with respect to all the others, and add it into the set.

An important point is that the polynomials in the Gröbner basis G are formed by adding or subtracting multiples of polynomials from F. This means every root of F is also a root of G, as we claimed above.

We apply Gröbner Basis techniques to solve enzyme kinetic problems and demonstrate the use of the techniques as a powerful tool in enzyme kinetic analysis.

2.1.2 Calculation of a Gröbner Basis

The REDUCE commend **groebner** calculates the Gröbner basis of the given set of expressions with respect to the given set of variables in the order given. We can try out Buchberger's algorithm in REDUCE, using the syntax

$$groebner(\{p_1, p_2, \ldots, p_n\}, \{v_1, v_2, \ldots, v_n\});$$

The first argument is a list of polynomials, the second a list of the variables. If the variable list is omitted, the variables in the polynomial list are used, ordered according to the system variable order, i.e. each additional polynomial of higher order due to the presence of a higher order variable or a higher power in that variable. A Gröbner basis $\{1\}$ means that the ideal generated by the input polynomials is the whole polynomial ring, or equivalently, that the input polynomials have no zeros in common.

The Buchberger is sometimes likened to the process of Gaussian elimination of linear equations. This is a little misleading, as it is a great deal more powerful, but it is true that for linear polynomials the algorithm is exactly Gaussian elimination; the S-polynomial is a linear combination that removes the leading monomial, and with a lexicographic order, we get a triangular system.

Chapter 3

Computing Methods

Since computers were invented, one of the main uses of electronic computers has been the solution of problems in mathematics. In the early days the lack of sophistication of existing hardware forced programmers to go to great lengths to produce software capable of solving difficult problems. The legacy of these pioneering efforts is still with us today in the structure of languages like FORTRAN, and the packaging of algorithms into libraries of subprograms.

We use computers here to solve enzyme kinetic problems. To do this, we perform all computer modelling using a SUN-4/260 using REDUCE [Hearn, 1987] and NAG [NAG, 1988] routines which were written in FORTRAN. We usually use the GENTRAN [Gates, 1987] package to generate the resulting formulae from REDUCE as FORTRAN expressions for numerical evaluation.

3.1 Computer Algebra Systems

Computers are usually used to manipulate numbers. However they can just as well work with other symbols, for example, algebraic variables. Computer algebra systems manipulate symbols not numbers. Rather than using the approximation methods of numerical analysis, they use exact algebraic techniques. Such systems tend to be interactive programs, commonly written in some dialect of LISP, and they accept their input in a quasi-mathematical notation which is simple to use and remember. They

can give general expressions as an answer, rather than only a numerical value. All the details of the algorithms used are normally hidden from the user, and the statement of the problem consists only of its logical, mathematical components.

Unfortunately computer algebra systems can be comparatively slow, and the time taken to solve a given problem is unpredictable, as is the size of the solution.

Computer algebra systems normally deal with the rational numbers and their polynomial extensions. There are generally no limits to the size of integers which the system can handle, and so any rational number can be represented exactly. Computer algebra systems also offer floating point arithmetic and, because the operations can be carried out in software rather than in hardware, they can offer arbitrarily high precision, specified at runtime. Numerical algorithms implemented in computer algebra systems and using floating point will generally run far slower than those implemented in FORTRAN libraries. It is perfectly possible to implement these algorithms in such a way as to use exact rational arithmetic but this leads to a problem:

- Comparing the relative size of two rational numbers whose denominators may be extremely large integers is not easy.

3.1.1 The Reduce Computer Algebra System

This system has been under development since the late 1960*s*, and the first version appeared in 1967 [Davenport, Siret and Tournier, 1988]. It was implemented in LISP, and it has continued to develop with the far-flung collaboration of its community of "advanced" users, who have contributed many modules and facilities to the system. The principal possibilities which this system offers are:

- Integer and rational "arbitrary precision" arithmetic.
- Machine and "arbitrary precision" floating point arithmetic.
- Polynomial algebra in one or several variables.
  - G.C.D. computations.
  - Factorisation of polynomials with integer coefficients.

- Matrix algebra, with polynomial or symbolic coefficients.
  - Determinant calculations.
  - Inverse computation.
  - Solution of systems of linear equations
- Calculus
  - Differentiation
  - Integration
- Manipulation of expressions
  - Simplification
  - Substitution

Although output is normally a "three-line" (floating point expression, rational expression and modular-coefficients integer expression) mathematical style, it can be generated as REDUCE input syntax and FORTRAN expressions.

REDUCE offers two *models* of operation: *algebraic* and *symbolic.* System programs are written in the latter, while applications programs are written in the former. Although REDUCE is untyped, it has the concept of a *domain.* The domain can be set by the user, and currently includes rational, rounded, Gaussian integer, complex rational, and complex rounded. Operations are performed in the current domain, and constants from other domains coerced to this when necessary. The rounded domain consists of floating point numbers. Where the precision is small enough, these are represented as hardware floats, otherwise they are bigfloats.

REDUCE offers a sophisticated package, GENTRAN for generating and optimising code in FORTRAN, which is discussed in detail in next section. The package GROEBNER calculates Gröbner Bases. An example is given in figure 3-1 and 3-2 to calculate Gröbner basis.

```
% A  Reduce program to calculate groebner basis for given polynomials.

% Polynomials.

f1 := 3*x^2*y+x^3*y+x*y+7*x+9*y;
f2 := x^2*y^2+x*y+x^5;

% load package.

load!-package 'groebner$

% Calculate groebner basis.

bas := groebner({f1,f2},{x,y});

end;
```

Figure 3-1: A REDUCE program to calculate Gröbner basis.

```
                                                  10                    9
BAS := {38186116692988019*X - 38652654787263*Y   + 103468614250235*Y

                              8                     7
        + 1497916898377853*Y  - 6401994256925805*Y

                              6                      5
        + 9068987272309151*Y  + 14982236331758037*Y

                               4                      3
        - 30364454208348212*Y  + 40189963463081887*Y

                               2
        - 10193824346531287*Y  + 49096435748127453*Y,

          11    10        9        8         7         6         5
       9*Y   - Y   - 409*Y  + 548*Y  + 1690*Y  - 6806*Y  - 6120*Y

              4         3         2
        + 8085*Y  - 1372*Y  - 2401*Y }
```

Figure 3-2: Result of the REDUCE program to calculate Gröbner basis.

3.1.2 Use of Computer Algebra System

The sceptic naturally asks: "What use are such complicated formulae" ? Other than using them as wall-paper, human beings do not want to look at expressions of this size. One major use of such formulae is numerical computation: REDUCE, like many algebra systems, can convert such expressions that can be incorporated into numerical programs. We have used REDUCE in six ways

1. We have used its matrix operations to implement Reder's algorithm for determining conservation rules.

2. We have used its linear algebra facilities to express all metabolite concentrations in terms of two independent concentrations.

3. We have used a Gröbner Basis calculation to obtain a steady state rate law for the complete system [Buchberger, 1985].

4. We have used it to evaluate FORTRAN expressions directly using an interfacing program written in C.

5. We have used it to evaluate higher derivatives of rate laws.

6. We have used the GENTRAN package to generate the resulting formulae as FORTRAN expressions for numerical evaluation [Gates, 1987].

3.2 Numerical Methods

As computers became more widespread and accessible to ordinary users, it became clear that much effort was being duplicated as individuals wrote their own implementations of well-known algorithms for numerical analysis, statistics, data sorting and so on. This led to the concept of a *subprogram library* which contained high-quality implementations of various algorithms which the users could link into their program during compilation. Although libraries were produced in a wide variety of languages, the most common was FORTRAN.

The advantages of subprogram libraries are clear: they provide efficient, reliable, and thoroughly tested pieces of code. However there are disadvantages as well. Although FORTRAN is fast, it is a very unintuitive language. Even for an experienced programmer, it can take some time to write and test the code to solve a relatively trivial problem. Not only do they need to translate it from its mathematical definition to its FORTRAN one, the user must frame it in the way required by the particular routine. Many conceptually simple operations are error-prone when done by hand: for example writing a subroutine to return the Jacobian of a given set of equations in an array. FORTRAN syntax is very low-level when compared with that of modern programming languages, and the restriction of parameter names to six characters prevents them from being either meaningful or memorable.

Another problem with FORTRAN is the amount of "unnecessary" information which the user must provide. Arrays for workspace, whose size depends in some way on the particular problem being solved, and array dimensions must all be explicitly passed as parameters to the routine. Because of its call-by-reference semantics the names of all parameters used to output results must also be passed. The correct ordering of the parameters is imperative, and since routines typically take not less than a dozen parameters (and often thirty or more) it is in practice impossible to use libraries without access to fairly detailed documentation, either printed or online.

Another aspect of using these "canned" algorithms is that a user needs a little understanding, not only of the problem being solved, but also of the method being used to solve it. This is because the routine will typically require a number of parameters to control the operation and termination of the algorithm: the maximum number of iterations to take in a quadrature routine, the step size to be used in finding the zero of a polynomial, or the accuracy required for the solution of a differential equation. The latter example also requires some knowledge of the precision of the implementation being used, and how well the algorithm may be expected to perform under the circumstances.

There is also the problem of choosing which algorithm to use in the first place, and this may involve some higher level of mathematical knowledge. For example a user wishing to solve a set of differential equations may have to decide whether it is stiff

or not, while someone using a quadrature routine might need to determine whether the integrand has any singularities or discontinuities. Advice giving systems, whether printed decision trees or interactive menu-driven systems, all tend to expect the user to understand such concepts, and be able to apply them to the particular problem in hand.

3.2.1 The NAG FORTRAN Library

The models studied in this thesis are analysed using a particular numerical analysis library: The NAG FORTRAN library [NAG ltd, 1990]. The library was started in 1970 and has grown steadily ever since. Many routines in the library were written in FORTRAN-IV at a time when the efficient use of memory was a far more critical consideration than it is in modern computing environments. This is reflected in the way data is handled and packed, and the way arrays are often used for multiple purposes in different situations.

The library is divided into chapters, each of which contains routines for solving particular problems. We used it for optimisation, solving cubic equations and integration.

The library contains around 1000 user callable routines [Dewar, 1991]. The routines lead to a great deal of inconsistency in the interfaces, particularly in the naming and style of parameters. (i.e, a subprogram argument which fulfils the same role in two subprograms may be of a completely different-and hence incompatible-format in two routines which solve the same problem).

The only fairly consistent part of the interfaces is the way in which they handle errors. There is a parameter, IFAIL, which we set to either -1, 0, or 1 before entering the routine. This determines whether, on encountering a fatal error, the routine will:

- hand control back to the calling program with a printed message (noisy soft fail),
- terminate the program with a printed message (hard fail),
- hand control back to the calling program without a message (silent soft fail).

If one of the soft fail options is chosen then, on exit, the value of IFAIL will have been set to an integer value which indicates what sort of error has occurred. The interpretation of these values is given in the printed and online documentation. Thus when the soft fail option is chosen it is always vital for the calling program to check the value of IFAIL on exit from a Library routine.

3.3 Combining Symbolic and Numeric Methods

Multiple enzyme kinetic problems benefit from a combination of both the symbolic and numerical approach. For example it is often convenient to model a problem symbolically, deriving a family of equations which describe it, and then evaluate those equations with various numerical values. The point here is that the two approaches—symbolic and numeric—should not be viewed as divorced from one another, but rather as complementary tools for problem solving. We will discuss two methods to combine symbolic and numeric methods in next section.

3.3.1 Generation of FORTRAN Expressions

Most computer algebra systems have some rudimentary method for producing output in FORTRAN. For example, in REDUCE we can set the switch *FORT* and all output will be converted to FORTRAN-compatible syntax. REDUCE's system is comparatively sophisticated; standard FORTRAN compilers will only accept a certain number of continuation lines, so it will only segment an expression which is too large, and then split it into several statements.

In this thesis, we usually used the GENTRAN package [Gates, 1987] to generate complete programs in FORTRAN, rather than just isolated expressions. Not only can GENTRAN translate most LISP statements, but the user may build skeletal programs, or *templates*, which are then "fleshed-out" by REDUCE. The *passive* parts of the template contain fragments of code in the target language, and are echoed verbatim to the output, while the *active* parts consist of sequences of REDUCE or GENTRAN commands. GENTRAN has a separate file-handling system, and so output from its

functions may be redirected to selected files, while the results of REDUCE functions still appear on the screen. There are also facilities for handling type declarations, and segmentation. In the latter case automatically-generated temporary variables are automatically declared to be of a default type (usually REAL).

Other facilities have been added to the original version of GENTRAN, including the handling of double precision constants and variables, the coercion of the arguments of FORTRAN intrinsic functions to the correct type, and the generation of complex constants [Dewar, 1991].

We want to give a simple example which gives a flavour of some of its facilities. Suppose that we wish to generate a FORTRAN function which will return a polynomial evaluated at a point. In figure 3-3 we see a GENTRAN template to perform this task. The `;BEGIN; ... ;END;` sequences enclose the active parts. Since we cannot tell in advance whether GENTRAN will generate any extra variables (to reduce expressions to the size allowed by the FORTRAN compiler), we produce the final program segment in two phases. Our template creates a second GENTRAN template whose only active part generates the symbol table then processed to get the final FORTRAN code. The sequence of steps needed in Reduce to do this is shown in figure 3-4, and the final result in figure 3-5. In practice, this multiple-pass technique is almost always necessary when translating expressions into FORTRAN.

3.3.2 Interface Between Computer Algebra and Numerical Analysis System

We commonly wish to solve enzyme kinetic problems using a mixture of analytical and numerical techniques. An effective approach is to use a computer algebra system to perform the analytical stage (solution of simultaneous equations, substitution of variables, determination of higher derivatives and so on) and then to use a standard numerical analysis package for the numerical stage (solution of differential equations, fitting to experimental data and so on). Numerical analysis packages are invariably written in languages such as FORTRAN and Pascal, with a rather different input syntax to that of the computer algebra system. To achieve an interface it is necessary

```
C       A GENTRAN Templete to generate a double precision function to
C       return the value of a multivariate polynomial at a given point.

        DOUBLE PRECISION FUNCTION  FUNC( X, Y, Z )
C
        REAL*8     X, Y, Z
C
;BEGIN;
        GENTRAN FUNC :=: F ;
;END;
C
        RETURN
        END
C
```

Figure 3-3: A GENTRAN Template.

```
% Reduce program to generate a double precision function.

 f := ( x + y + z )^6 ;

% Put it out in GENTRAN

load!-package 'gentran $
gentranlang!*:= 'FORTRAN $
fortlinelen!*:= 71 $

on numval $
on double $

gentranin ''example.tem'' out ''example.f'' $
%
end $
```

Figure 3-4: A Reduce program using GENTRAN.

```
C       A GENTRAN TEMPLETE TO GENERATE A DOUBLE PRECISION FUNCTION TO
C       RETURN THE VALUE OF A MULTIVARIATE POLYNOMIAL AT A GIVEN POINT.

        DOUBLE PRECISION FUNCTION  FUNC( X, Y, Z )
C
        REAL*8     X, Y, Z
C
      FUNC=X**6+6.0D0*X**5*Y+6.0D0*X**5*Z+15.0D0*X**4*Y**2+30.0D0*X**4*
     . Y*Z+15.0D0*X**4*Z**2+20.0D0*X**3*Y**3+60.0D0*X**3*Y**2*Z+60.0D0*
     . X**3*Y*Z**2+20.0D0*X**3*Z**3+15.0D0*X**2*Y**4+60.0D0*X**2*Y**3*Z
     . +90.0D0*X**2*Y**2*Z**2+60.0D0*X**2*Y*Z**3+15.0D0*X**2*Z**4+6.0D0
     . *X*Y**5+30.0D0*X*Y**4*Z+60.0D0*X*Y**3*Z**2+60.0D0*X*Y**2*Z**3+
     . 30.0D0*X*Y*Z**4+6.0D0*X*Z**5+Y**6+6.0D0*Y**5*Z+15.0D0*Y**4*Z**2+
     . 20.0D0*Y**3*Z**3+15.0D0*Y**2*Z**4+6.0D0*Y*Z**5+Z**6
C
        RETURN
        END
```

Figure 3-5: An example of FORTRAN generated by GENTRAN.

for the results from the computer algebra system to be output in a suitable form—FORTRAN expressions for example. Most computer algebra systems now provide fairly sophisticated packages to do this, for example REDUCE's GENTRAN package [Hearn, 1987; Gates, 1987]).

There is a problem with this approach in that the computer algebra system may generate massive expressions. On evaluation with specific values in floating point by the numerical analysis package such expressions are prone to serious rounding, overflow and underflow errors. However we demonstrate an alternative approach, where we get the computer algebra system to evaluate the expressions analytically for the numerical analysis package as and when they are needed.

We demonstrate the use of this technique to solve enzyme kinetics problem using REDUCE and the NAG FORTRAN library under the Unix operating system. However the technique is completely general and can essentially be applied to any computer algebra system and numerical analysis package working under a multi-tasking operating system. The technique will be given in following section.

3.3.3 Driver Code for Interfacing

We start by running a REDUCE session in parallel with our FORTRAN numerical analysis program. However we arrange to connect the standard input of each to the standard output of the other. Thus the REDUCE session reads commands that have been written out by the FORTRAN program, and the FORTRAN program can read the results which REDUCE prints out.

Figure 3-6 shows the code which sets up this system, which for convenience under Unix is written in the C programming language. For simplicity we have stripped out of this code any system error checking.

We start by creating two pipes. These are like files, but are used for communicating between processes and we need one for each direction. The pipes are a vector of 2 elements each. Element 0 is for reading and element 1 is for writing. We then use `fork()` to create two processes, one running REDUCE, the other a FORTRAN program, `fortprog`. These are started in the two routines `reduce` and `fortran`. To each of these routines we pass one end of each pipe, and these are then mapped on to the standard input and output streams using `dup2`. Finally REDUCE and `fortprog` are started, inheriting these pipes as their standard input and output.

It is convenient to view `fortprog` as the driving routine. It issues commands to REDUCE, and then reads back the result. An example is shown in figure 3-7 which just calls REDUCE to evaluate an expression. We typically start by bringing in some initialisation code, with the REDUCE command `in "test.red" $`. This will do various initial calculations including defining the expression(s) to be evaluated later. To ensure this command is sent to REDUCE and not buffered by the I/O system we call the system routine `FLUSH`. Finally we call the (user written) routine `NEXTC` to skip past the command prompt which will occur after each REDUCE command. This simply reads in lines until one ending in `:␣` is encountered.

Evaluation of expressions involves issuing a suitable REDUCE command. In the example above we use `sub( x = i, y = j, ratelaw )` with the values of the FORTRAN variables `I` and `J` substituted for `i` and `j`. We again call `FLUSH` to ensure this is sent, `CALL NEXTC` to skip the command prompt and then read in the value REDUCE

```
void  main()
{
        int  atob[2] ;
        int  btoa[2] ;

        pipe( atob ) ;                          /* Create the pipes */
        pipe( btoa ) ;

        if( fork() == 0 )
                reduce( atob[0], btoa[1] ) ;    /* Child runs REDUCE */
        else
                fortran( btoa[0], atob[1] ) ;   /* Parent runs FORTRAN */

}       /* main( void ) */


void reduce( int  data_in,
             int  data_out )
{
        char *argv[] = { "reduce3.4", NULL } ;  /* Argument vector */
        char *envp[] = { NULL } ;               /* Environment vector */

        dup2( data_in, FD_IN ) ;                /* Change file descriptors */
        dup2( data_out, FD_OUT ) ;

        execve( "/usr/local/bin/reduce3.4", argv, envp ) ; /* Run REDUCE */

}       /* reduce( data_in, data_out ) */


void fortran( int  data_in,
              int  data_out )
{
        char *argv[] = { "fortprog", NULL } ;   /* Argument vector */
        char *envp[] = { NULL } ;               /* Environment vector */

        dup2( data_in, FD_IN ) ;                /* Change file descriptors */
        dup2( data_out, FD_OUT ) ;

        execve( "fortprog", argv, envp ) ;      /* Run the FORTRAN program */

}       /* fortran( data_in, data_out ) */
```

Figure 3-6: Driver code for the parallel processes.

```
C
C     Initialise REDUCE
C
      WRITE( 6, 99990 )
      CALL FLUSH( 6 )
      CALL NEXTC
C
C     Evaluate an expression and read back the result
C
      WRITE( 6, 99980 ) I, J
      CALL FLUSH( 6 )
C
      CALL NEXTC
      READ( 5, * ) VALUE
C
C     Leave REDUCE tidily
C
      WRITE( 6, 99970 )
      CALL FLUSH( 6 )
C
      STOP
C
99990 FORMAT( 'in "test.red" $' )
99980 FORMAT( 'sub( x = ', I3, ', y = ', I3 ', ratelaw ) ;' )
99970 FORMAT( 'bye ;' )
      END
```

Figure 3-7: FORTRAN code to call REDUCE.

prints as result.

This may not be the ideal arrangement unless the results are integers. Rational results may be read in as separate numerator and denominator, or the rational may then be evaluated using `ON ROUNDED` to get a floating point result. In this case `PRECISION 15` will prove convenient as corresponding to the maximum precision for 64 bit IEEE floating point numbers.

Finally we can issue a `bye` command to REDUCE to ensure the process shuts down tidily.

3.3.4 Summary

For a given problem, a little careful thought may dramatically inprove the efficiency of the code being generated. Existing libraries of FORTRAN subprograms represent decades of effort, and have proved themselves to be robust and reliable. Unfortunately they are difficult to use, and do not fit comfortably into the modern interactive view of computer software. Computer algebra systems are friendly, interactive packages which, with the growth of computer power available to most users, are deservedly becoming more popular.

The two approaches are useful in different circumstances, but can be combined very effectively to tackle a number of classes of problems. Clearly any problem-solving toolkit for the modern scientist should include both symbolic and numerical facilities. We have improved a interfacing technique between Computer algebra system and numerical analysis packages. The technique is completely general and can essentially be applied to any computer algebra system and numerical analysis package working under a multi-tasking operating system.

Chapter 4

Analysis of Coupled Systems

An ability to analyse a system of multiple of enzymes simultaneously would be a great advantage in kinetic modelling. The subject of this section is an analysis of the feasibility of applying computer algebra to kinetics of multienzyme systems, and ultimately to analysis of such systems in *vivo*. The steady state approach, whereby reaction rates of enzymes in a linear pathway may be linked, is identified as a candidate for computer analysis using Gröbner bases, with potential commercial ramifications. In this chapter, first we will give a brief review of chemical kinetics that should serve to reacquaint us with the theoretical basis of our approach to enzyme analysis. Secondly, experimental procedures will be given for coupled and single enzyme systems that have been provided by D.L. Fisher [Fisher, 1990a-b]. We have used a single experiment for every coupled and isolated system Thirdly, a conventional simulation of the quantitative behaviour of the metabolites in the enzyme system will be performed.

4.1 Kinetic Theory

Enzyme kinetic systems are usually defined at any point in space and time by three vectors: a state variable vector; a parameter vector and a boundary (initial) conditions vector. The state variables are certain observable macroscopic quantities and are defined as the minimum set of variables necessary to describe the present and future state of the system; examples include temperature, enzyme and metabolite concentrations. The

parameter vector includes properties of the system that are independent of the state of the system and examples include rate constants and fixed enzyme concentrations. Finally, the boundary conditions vector includes all external forces that could possible affect the system and examples may include light intensity, external concentrations and hormones.

The enzyme kinetic systems are expressed as a set of differential equations [Garfinkel, 1963] where each state variable has associated with it a single differential equation describing its motion in both space and time. We analyse such systems using computer algebra techniques.

4.1.1 Steady State Approximation

If an intermediate is always present in amounts much less than those of the reactants (other than the enzyme) the rate of change of its concentration is much smaller than that of the reactants. This condition is ensured whenever, as is usually the case in enzyme catalysed reactions *in vitro* [Albe and Wright 1990], the concentration of substrate is much higher than that of the enzyme; it is not necessary for the amount of intermediate to be small compared with the amount of enzyme [Engel, 1981]. For example, consider the scheme

$$\mathrm{A} + \mathrm{E} \underset{k_2}{\overset{k_1}{\rightleftharpoons}} \mathrm{EA} \underset{k_4}{\overset{k_3}{\rightleftharpoons}} \mathrm{P} + \mathrm{E}$$

If EA is always much less than [A] the following equation, to a good approximation, is satisfied.

$$\frac{d[\mathrm{EA}]}{dt} = k_1 \cdot [\mathrm{A}] \cdot [\mathrm{E}] - (k_3 + k_2) \cdot [\mathrm{EA}] + k_4 \cdot [\mathrm{P}] \cdot [\mathrm{E}] = 0. \tag{4.1}$$

The intermediate, EA, is said to be in a steady state. The use of this approximation to obtain an overall rate expression is known as the *Steady State Assumption* or *Steady State Approximation.* We use this assumption to obtain steady state rate laws in enzyme kinetics.

4.1.2 Formulation of Reactions

In this thesis we look at the three enzyme system of aspartate aminotransferase, malate dehydrogenase and fumarase. In this section, we will give a formulation of their reactions apart from the fumarase mechanism, because that formulation will be given in the next section.

The general formulation of the aspartate aminotransferase is based upon the following observations [Velick and Vavra, 1962].

1. The bound coenzyme undergoes cyclic interconversions between the amino and aldehyde forms.

2. The enzyme catalyses amino group exchange between glutamate and α-ketoglutarate in the total absence of the aspartate oxaloacetate pair.

These results indicate that the reversible amino group transfer between aspartate and α-ketoglutarate is the sum of two half reactions such as;

$$\begin{aligned} aspartate + aldehydenzyme &\rightleftharpoons oxaloacetate + aminoenzyme \\ aminoenzyme{+}\alpha{-}ketoglutarate &\rightleftharpoons glutamate + aldehydenzyme \end{aligned}$$

The second enzyme mechanism is a special case of the compulsory binding order mechanism in which a ternary complex is formed and decomposes so rapidly that it escapes detection by ordinary initial rate kinetic studies [Raval and Wolfe, 1962]. This reaction mechanism is represented by

$$\begin{aligned} E + NADH &\rightleftharpoons E.NADH \\ E.NADH + oaa &\rightleftharpoons E.NADH.oaa \\ E.NADH.oaa &\rightleftharpoons E.NAD + mal \\ E.\mathrm{NAD}^{+} &\rightleftharpoons E{+}\mathrm{NAD}^{+} \end{aligned}$$

The abbreviations used here include: NAD^{+} for coenzyme form of nicotinamide adenine dinucleotide, $NADH$ for reduced form of NAD^{+} and E for enzyme.

4.1.3 Derivation of Steady State Rate Laws

To obtain the rate law for a specific mechanism one first has to define the differential equations and the enzyme conservation equation that describe the mechanism. The differential equations are written in terms of the kinetic constants of the elementary reactions of the mechanism k_i and k_{-i}. Assuming a steady-state, these form a system of ordinary equations which is solved using the mathematical technique of Gröbner Bases [Buchberger, 1985]. The result is the rate equation expressed in terms of the k_i's. Rate equations in this form may not be very useful if they are going to be used as models for experimental enzyme kinetics. In that case one would like to have these equations expressed with more convenient parameters such as K_M or V_{max}'s. There is, however, more than one set of such parameters in multisubstrate multiproduct reactions due to the appearence of crossed terms. An indication of the automatic derivation of the steady state rate equations will be given in chapter 7.

4.1.4 Aspartate Aminotransferase

The general formulation of the mechanism was given in the previous section by

$$\text{asp} + \text{E} \underset{k_2}{\overset{k_1}{\rightleftharpoons}} \text{E.asp} \underset{k_4}{\overset{k_3}{\rightleftharpoons}} \text{oaa} + E'$$

$$\alpha\text{-kg} + E' \underset{k_6}{\overset{k_5}{\rightleftharpoons}} E'.\alpha\text{-kg} \underset{k_8}{\overset{k_7}{\rightleftharpoons}} \text{glu} + \text{E}.$$

The symbols asp, α-kg, glu and oaa refer to concentrations of aspartate, α-ketoglutarate, glutamate and oxaloacetate, respectively. E and E' are the functionally non-dissociable complexes of pyridoxal and pyridoxamine phosphate with enzyme, and E.asp and $E'.\alpha$-kg are symbols for sequences of intermediates.

In terms of the rate constants shown, the steady-state equations for all the intermediates as follows;

$$
\begin{aligned}
\partial[\mathbf{E.asp}]/\partial\mathbf{t} &= k_1 \cdot [asp] \cdot [E] - (k_2 + k_3) \cdot [E.asp] + k_4 \cdot [oaa] \cdot [E'] = 0 \\
\partial[\mathbf{E'}]/\partial\mathbf{t} &= k_3 \cdot [E.asp] + k_6 \cdot [E'.\alpha\text{-kg}] \cdot -k_4 \cdot [E'] \cdot [oaa] - k_5 \cdot [E'] \cdot [\alpha\text{-kg}] = 0 \\
\partial[\mathbf{E'.\alpha\text{-}kg}]/\partial\mathbf{t} &= k_5 \cdot [E'] \cdot [\alpha\text{-kg}] + k_8 \cdot [E] \cdot [glu] - (k_6 + k_7) \cdot [E'.\alpha\text{-kg}] = 0
\end{aligned}
\tag{4.2}
$$

The total concentration of enzyme is

$$
E_0 = (E + E.asp + E^{'}.\alpha\text{-kg} + E^{'}). \tag{4.3}
$$

The rate of production of glu which is called the *rate of the reaction* is

$$
v_1 = \partial[\mathbf{glu}]/\partial\mathbf{t} = k_7 \cdot [E^{'}.\alpha\text{-kg}] - k_8 \cdot [glu] \cdot [E] \tag{4.4}
$$

i.e.

$$
v_1 - (k_7 \cdot [E^{'}.\alpha\text{-kg}] - k_8 \cdot [glu] \cdot [E]) = 0 \tag{4.5}
$$

The solution of the above five equations was done using Gröbner bases. All the intermediates, [E.asp], [E], [$E^{'}$], [$E^{'}.\alpha$-kg], are eliminated to yield a formulae for v_1. Finally the following rate equation is obtained:

$$
\begin{aligned}
v_1 = {} & (E_0 \cdot (k_1 \cdot [asp] \cdot k_3 \cdot k_5 \cdot \alpha\text{-kg} \cdot k_7 - k_2 \cdot k_4 \cdot [oaa] \cdot k_6 \cdot k_8 \cdot [glu]))/ \\
& (k_1 \cdot [asp] \cdot k_3 \cdot k_6 + k_1 \cdot [asp] \cdot k_3 \cdot k_5 \cdot akg + k_1 \cdot [asp] \cdot k_3 \cdot k_7 \\
& + k_1 \cdot [asp] \cdot k_4 \cdot [oaa] \cdot k_6 + k_1 \cdot [asp] \cdot k_4 \cdot [oaa] \cdot k_7 \\
& + k_1 \cdot [asp] \cdot k_5 \cdot \alpha\text{-kg} \cdot k_7 + k_2 \cdot k_4 \cdot [oaa] \cdot k_6 + k_2 \cdot k_4 \cdot [oaa] \cdot k_8 \cdot [glu] \\
& + k_2 \cdot k_4 \cdot [oaa] \cdot k_7 + k_2 \cdot k_6 \cdot k_8 \cdot [glu] + k_2 \cdot k_5 \cdot \alpha\text{-kg} \cdot k_8 \cdot [glu] \\
& + k_2 \cdot k_5 \cdot \alpha\text{-kg} \cdot k_7 + k_3 \cdot k_6 \cdot k_8 \cdot [glu] + k_3 \cdot k_5 \cdot \alpha\text{-kg} \cdot k_8 \cdot [glu] \\
& + k_3 \cdot k_5 \cdot \alpha\text{-kg} \cdot k_7 + k_4 \cdot [oaa] \cdot k_6 \cdot k_8 \cdot [glu])
\end{aligned}
\tag{4.6}
$$

The above equation can be rewritten in the following form:

$$v_1 = \frac{\frac{V^F_{max}[\text{asp}][\alpha\text{-kg}]}{K_I^{asp} \cdot K_M^{\alpha\text{-kg}}} - \frac{V^R_{max}[\text{oaa}][\text{glu}]}{K_I^{oaa} \cdot K_M^{glu}}}{\frac{[\text{asp}]}{K_I^{asp}} + \frac{K_M^{asp}[\alpha\text{-kg}]}{K_I^{asp} \cdot K_M^{\alpha\text{-kg}}} + \frac{[\text{oaa}]}{K_I^{oaa}} + \frac{K_M^{oaa}[\text{glu}]}{K_I^{oaa} \cdot K_M^{glu}} + \frac{[\text{asp}][\alpha\text{-kg}]}{K_I^{asp} \cdot K_M^{\alpha\text{-kg}}} + \frac{[\text{asp}][\text{oaa}]}{K_I^{asp} \cdot K_I^{oaa}} + \frac{K_M^{asp}[\alpha\text{-kg}][\text{glu}]}{K_I^{asp} \cdot K_M^{\alpha\text{-kg}} K_I^{glu}} + \frac{[\text{oaa}][\text{glu}]}{K_I^{oaa} \cdot K_M^{glu}}} \tag{4.7}$$

where

$$\begin{aligned}
V^F_{max} &= (k_3 \cdot k_7 \cdot [E_0])/(k_3 + k_7) \\
V^R_{max} &= (k_2 \cdot k_6 \cdot [E_0])/(k_2 + k_6) \\
K_M^{asp} &= ((k_2 + k_3) \cdot k_7)/(k_1 \cdot (k_3 + k_7)) \\
K_M^{\alpha\text{-kg}} &= (k_3 \cdot (k_6 + k_7))/((k_3 + k_7) \cdot k_5) \\
K_M^{oaa} &= ((k_2 + k_3) \cdot k_6)/((k_2 + k_6) \cdot k_4) \\
K_M^{glu} &= (k_2 \cdot (k_6 + k_7))/((k_2 + k_6) \cdot k_8) \\
K_I^{asp} &= k_2/k_1 \\
K_I^{\alpha\text{-kg}} &= k_6/k_5 \\
K_I^{oaa} &= k_3/k_4 \\
K_I^{glu} &= k_7/k_8.
\end{aligned} \tag{4.8}$$

Symbols K_M^{asp}, $K_M^{\alpha\text{-kg}}$, K_M^{oaa} and K_M^{glu} are the Michaelis constants of aspartate, α-ketoglutarate, oxaloacetate and glutamate (the concentration of *asp*, α-kg, *oaa* and *glu* at which the reaction proceeds at half the velocity V_{max}), respectively. K_I^{asp}, $K_I^{\alpha\text{-kg}}$, K_I^{oaa} and K_I^{glu} are the inhibition constant of aspartate, α-ketoglutarate, oxaloacetate and glutamate, respectively. The inhibition constants represent the metabolite's tendency to bind the enzyme at the wrong stage, inhibiting its activity. V_{max} is maximum velocity which depends on the concentration of the enzyme, the scripts F and R, refer to forward and reverse reaction directions.

Although equation 4.7 does not contain $K_I^{\alpha\text{-kg}}$ it can be rewritten so that it does by means of the identity $K_I^{asp} \cdot K_M^{\alpha\text{-kg}}/K_I^{oaa} \cdot K_M^{glu} = K_M^{asp} \cdot K_I^{\alpha\text{-kg}}/K_M^{oaa} \cdot K_I^{glu}$ [Cornish-Bowden , 1979].

4.1.5 Malate Dehydrogenase

The mechanism is a compulsory binding order process. There are eight rate constants for the four reversible reactions. The reaction mechanism such as

$$\text{E} + \text{NADH} \underset{k_2}{\overset{k_1}{\rightleftharpoons}} \text{E.NADH}$$

$$\text{E.NADH} + \text{oaa} \underset{k_4}{\overset{k_3}{\rightleftharpoons}} \text{E.NADH.oaa}$$

$$\text{E.NADH.oaa} \underset{k_6}{\overset{k_5}{\rightleftharpoons}} \text{E.NAD} + \text{mal}$$

$$\text{E.NAD} \underset{k_8}{\overset{k_7}{\rightleftharpoons}} \text{E} + NAD^+$$

Here the NADH, oaa, mal and NAD^+ refer to concentrations of NADH, oxaloacetate, malate and NAD^+, respectively. $E.NADH$ and $E.NAD$ are binary complexes and $E.NADH.oaa$ is a ternary complex.

In terms of the rate constants shown, the steady-state equations for all the intermediates as follows:

$$\begin{aligned}
\partial[\mathbf{E.NADH}]/\partial\mathbf{t} &= k_1 \cdot [E] \cdot [NADH] + k_4 \cdot [E.NADH.oaa] \\
&\quad -k_2 \cdot [E.NADH] - k_3 \cdot [E.NADH] \cdot [oaa] = 0, \\
\partial[\mathbf{E.NADH.oaa}]/\partial\mathbf{t} &= k_3 \cdot [E.NADH] \cdot [oaa] + k_6 \cdot [E.NAD] \cdot [mal] \\
&\quad -(k_4 + k_5) \cdot [E.NADH.oaa] = 0 \\
\partial[\mathbf{E.NAD}]/\partial\mathbf{t} &= k_5 \cdot [E.NADH.oaa] + k_8 \cdot [E] \cdot [NAD] \\
&\quad -k_6 \cdot [E.NAD] \cdot [mal] - k_7 \cdot [E.NAD] = 0.
\end{aligned} \tag{4.9}$$

We also have another equation like equation 4.3 relating the four unknown variables, E, E.NADH, E.NADH.oaa, E.NAD. It is called the *enzyme conservation equation*, which

states in essence that we know how much enzyme was put in, although we may not know precisely where it has gone. Thus,

$$[E_0] - ([E] + [E.NADH] + [E.NADH.oaa] + [E.NAD]) = 0 \tag{4.10}$$

where $[E_0]$ is the total enzyme concentration. The rate of production of NAD^+ is

$$v_2 = \partial[\mathbf{NAD}]/\partial \mathbf{t} = k_7 \cdot [E.NADH.oaa] - k_8 \cdot [E] \cdot [NAD]. \tag{4.11}$$

Now we eliminate all the intermediates and free enzyme concentrations using Gröbner bases. Finally we obtain following equation.

$$\begin{aligned} v_2 \;=\; & ([E_0] \cdot (k_1 \cdot [NADH] \cdot k_3 \cdot [oaa] \cdot k_5 \cdot k_7 - k_4 \cdot k_2 \cdot k_6 \cdot [mal] \cdot \\ & k_8 \cdot [NAD]))/(k_1 \cdot [NADH] \cdot k_4 \cdot k_6 \cdot [mal] + k_1 \cdot [NADH] \cdot k_4 \cdot k_7 \\ & + k_1 \cdot [NADH] \cdot k_3 \cdot [oaa] \cdot k_6 \cdot [mal] + k_1 \cdot [NADH] \cdot k_3 \cdot [oaa] \cdot k_5 \\ & + k_1 \cdot [NADH] \cdot k_3 \cdot [oaa] \cdot k_7 + k_1 \cdot [NADH] \cdot k_5 \cdot k_7 + k_4 \cdot k_2 \cdot k_6 \cdot [mal] \\ & + k_4 \cdot k_2 \cdot k_8 \cdot [NAD] + k_4 \cdot k_2 \cdot k_7 + k_4 \cdot k_6 \cdot [mal] \cdot k_8 \cdot [NAD] \\ & + k_2 \cdot k_6 \cdot [mal] \cdot k_8 \cdot [NAD] + k_2 \cdot k_5 \cdot k_8 \cdot [NAD] + k_2 \cdot k_5 \cdot k_7 \\ & + k_3 \cdot [oaa] \cdot k_6 \cdot [mal] \cdot k_8 \cdot [NAD] + k_3 \cdot [oaa] \cdot k_5 \cdot k_8 \cdot [NAD] \\ & + k_3 \cdot [oaa] \cdot k_5 \cdot k_7) \end{aligned} \tag{4.12}$$

As can be seen, the above rate law is in terms of individual rate constants. If such forms of the rate laws are not useful, then we can obtain the rate laws in terms of Michaelis constants, $\mathbf{K_M}$ and $\mathbf{V_{max}}$, such as

$$\begin{aligned} v_2 = \; & \frac{\frac{V^F_{max}[\mathrm{NADH}][\mathrm{oaa}]}{K_I^{\mathrm{NADH}} \cdot K_M^{\mathrm{oaa}}} - \frac{V^R_{max}[\mathrm{mal}][\mathrm{NAD^+}]}{K_M^{\mathrm{mal}} \cdot K_I^{\mathrm{NAD^+}}}}{1 + \frac{[\mathrm{NADH}]}{K_I^{\mathrm{NADH}}} + \frac{K_M^{\mathrm{NADH}}[\mathrm{oaa}]}{K_I^{\mathrm{NADH}} \cdot K_M^{\mathrm{oaa}}} + \frac{K_M^{\mathrm{NAD^+}}[\mathrm{mal}]}{K_M^{\mathrm{mal}} \cdot K_I^{\mathrm{NAD^+}}} + \frac{[\mathrm{NAD^+}]}{K_I^{\mathrm{NAD^+}}}} \\ & + \frac{[\mathrm{NADH}][\mathrm{oaa}]}{K_I^{\mathrm{NADH}} \cdot K_M^{\mathrm{oaa}}} + \frac{K_M^{\mathrm{NAD^+}}[\mathrm{NADH}][\mathrm{mal}]}{K_I^{\mathrm{NADH}} \cdot K_M^{\mathrm{mal}} \cdot K_I^{\mathrm{NAD^+}}} + \frac{K_M^{\mathrm{NADH}}[\mathrm{oaa}][\mathrm{NAD^+}]}{K_I^{\mathrm{NADH}} \cdot K_M^{\mathrm{oaa}} \cdot K_I^{\mathrm{NAD^+}}} \\ & + \frac{[\mathrm{mal}][\mathrm{NAD^+}]}{K_M^{\mathrm{mal}} \cdot K_I^{\mathrm{NAD^+}}} + \frac{[\mathrm{NADH}][\mathrm{oaa}][\mathrm{mal}]}{K_I^{\mathrm{NADH}} \cdot K_M^{\mathrm{oaa}} \cdot K_I^{\mathrm{mal}}} + \frac{[\mathrm{oaa}][\mathrm{mal}][\mathrm{NAD^+}]}{K_I^{\mathrm{oaa}} \cdot K_M^{\mathrm{mal}} \cdot K_I^{\mathrm{NAD^+}}} \end{aligned} \tag{4.13}$$

where

$$
\begin{aligned}
V_{max}^{F} &= (k_5 \cdot k_7 \cdot [E_0])/(k_5 + k_7) \\
V_{max}^{R} &= (k_2 \cdot k_4 \cdot [E_0])/(k_2 + k_4) \\
K_M^{NADH} &= (k_5 \cdot k_7)/(k_1 \cdot (k_5 + k_7)) \\
K_M^{NAD} &= (k_2 \cdot k_4)/((k_2 + k_4) \cdot k_8) \\
K_M^{oaa} &= ((k_4 + k_5) \cdot k_7)/(k_3 \cdot (k_5 + k_7)) \\
K_M^{mal} &= (k_2 \cdot (k_4 + k_5))/((k_2 + k_4) \cdot k_6) \\
K_I^{oaa} &= (k_2 + k_4)/k_3 \\
K_I^{mal} &= (k_5 + k_7)/k_6 \\
K_I^{NADH} &= k_2/k_1 \\
K_I^{NAD} &= k_7/k_8.
\end{aligned}
\tag{4.14}
$$

The symbols K_M^{NADH}, K_M^{NAD}, K_M^{oaa} and K_M^{mal} are Michaelis constants for NADH, NAD, oxaloacetate and malate respectively. K_I^{NADH}, K_I^{NAD}, K_I^{oaa} and K_I^{mal} are inhibition constants for NADH, NAD, oxaloacetate and malate respectively. The symbol V_{max}^{F} denotes forward maximum velocity, and V_{max}^{R} is reverse maximum velocity.

4.1.6 Fumarase

The fumarase mechanism is

$$
\text{mal} + \text{E} \underset{k_2}{\overset{k_1}{\rightleftharpoons}} \text{E.mal} \underset{k_4}{\overset{k_3}{\rightleftharpoons}} \text{fum} + \text{E}.
$$

The steady state assumption of the mechanism is now expressed by

$$
\partial[\mathbf{E.mal}]/\partial \mathbf{t} = k_1 \cdot [E] \cdot [mal] - (k_2 + k_3) \cdot [E.mal] + k_4 \cdot [fum] \cdot [E] = 0, \tag{4.15}
$$

and the enzyme conservation equation is

$$
E_0 - (E + E.mal) = 0. \tag{4.16}
$$

The rate of production of fumarate (which is called the rate of reaction) is given by

$$v_3 = \partial[\mathbf{fum}]/\partial t = (k_3 \cdot [E.mal] - k_4 \cdot [fum] \cdot [E]. \tag{4.17}$$

The solution of the above three equations is done using Gröbner bases. All the intermediates, [E.mal] and [E], are eliminated. Finally the following rate equation is obtained.

$$v_3 = \frac{E_o \cdot [\text{mal}] \cdot k_1 \cdot k_3 - E_o \cdot [\text{fum}] \cdot k_2 \cdot k_4)}{[\text{fum}] \cdot k_4 + [\text{mal}] \cdot k_1 + k_2 + k_3} \tag{4.18}$$

The rate equation can be written in terms of Michaelis constants such as

$$v_3 = \frac{\frac{V^F_{max}[\text{mal}]}{K^{mal}_M} - \frac{V^R_{max}[\text{fum}]}{K^{fum}_M}}{1 + \frac{[\text{mal}]}{K^{mal}_M} + \frac{[\text{fum}]}{K^{fum}_M}} \tag{4.19}$$

The relationship between rate and Michaelis constants is

$$\begin{aligned} K^{mal}_M &= (k_2 + k_3)/k_1 \\ K^{fum}_M &= (k_2 + k_3)/k_4 \\ V^F_{max} &= k_3 \cdot [E_0] \\ V^R_{max} &= k_2 \cdot [E_0] \end{aligned} \tag{4.20}$$

where K^{mal}_M and K^{fum}_M are the Michaelis constants for malate and fumarate respectively. V^F_{max} and V^R_{max} are forward and reverse maximum velocities of the reaction.

4.2 Conservation Relationships

In the previous section, an enzyme kinetic system was described as a system of differential equations. In this section we will give more relationships – known as *metabolite conservation relationships* – between metabolite concentrations of the given enzyme kinetic systems. These are required for the elimination of the intermediate metabolite variables in the enzyme kinetic system.

Consider the enzyme kinetic system, whose mathematical representation we wish to

construct, and suppose that its state is entirely described by the following quantities: metabolite concentrations, pressure, temperature, etc.

Let us consider the aspartate aminotransferase-malate dehydrogenase biochemical system as example, of which variables are metabolite concentration; **asp, α-kg, glu, oaa, NADH, mal** and **NAD^+**. We define the metabolite concentration vector **X** as

$$X = \begin{bmatrix} asp \\ \alpha\text{-kg} \\ glu \\ oaa \\ NADH \\ mal \\ NAD^+ \end{bmatrix} \tag{4.21}$$

In order to construct the model, we first write the stoichiometric reaction scheme that describes how the metabolites combine [Reder, 1988].

It will be convenient to associate to the reaction scheme the matrix **N** of m rows and r columns constructed as follows: the column j of **N** represents the reaction j, and we write in this column at row i. For example, for metabolite **asp** and reaction 1, **N** (i, j)= **N**(1,1).

- $+\alpha$ if the reaction j produces α molecules of asp.
- $-\alpha$ if the reaction j consumes α molecules of asp.
- 0 if the reaction j neither produces nor consumes asp.

For the aspartate aminotransferase-malate dehydrogenase coupled system, the stoichiometric matrix is

$$\mathbf{N} = \begin{bmatrix} -1 & 0 \\ -1 & 0 \\ 1 & 0 \\ 1 & -1 \\ 0 & -1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} \tag{4.22}$$

We assume that the rate of change of the concentration of metabolite is the sum of the r reaction rates, each weighted by the corresponding stoichiometric coefficient of metabolites.

Let v_j denotes the rate of the reaction j, for aspartate aminotransferase-malate dehydrogenase:

$$v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \tag{4.23}$$

This is called the rate vector of the aspartate aminotransferase-malate dehydrogenase system. Now we may write

$$\frac{d}{dt}\begin{bmatrix} asp \\ \alpha\text{-kg} \\ glu \\ oaa \\ NADH \\ mal \\ \text{NAD}^+ \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ -1 & 0 \\ 1 & 0 \\ 1 & -1 \\ 0 & -1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \tag{4.24}$$

or equivalently as follows:

$$\frac{d}{dt}\begin{bmatrix} asp \\ \alpha\text{-kg} \\ glu \\ oaa \\ NADH \\ mal \\ \text{NAD}^+ \end{bmatrix} = \begin{bmatrix} -v_1 \\ -v_1 \\ v_1 \\ v_1 - v_2 \\ -v_2 \\ v_2 \\ v_2 \end{bmatrix} \tag{4.25}$$

The v_j are functions both of the m concentrations of metabolites. The conservation relationship of a given biochemical system is based on a decomposition of the scheme matrix **N** that we will now explain.

Let us extract from **N** a subset of its rows which constitutes a basis of the whole set of its rows. For such a subset:

1. The rows are linearly independent, which means that there is no null linear combination of these rows, except the trivial one where all the coefficients are zero.

2. Every row of **N** is a linear combination of the rows of this subset.

Whatever extraction method is used, the number of rows of such a subset must be the same. By definition, this number is called the *rank* of the matrix **N**, and we shall use m_o to denote it.

As it is always possible to change the order of the rows of **N**–that is to renumber the metabolites–we shall assume that its first m_o rows satisfy the above properties. Let $\mathbf{N_R}$ denote the matrix composed of those m_o rows. The properties we have imposed to them allow us to decompose **N** as the product:

$$\mathbf{N} = \mathbf{L} \cdot \mathbf{N_R} \tag{4.26}$$

where the m row and m_o column matrix $\mathbf{L}$ has the form :

$$\mathbf{L} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 & 0 \\ . & . & . & \cdots & . & . & . \\ 0 & 0 & 0 & \cdots & 0 & 1 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 1 \\ - & - & - & \cdots & - & - & - \\ & & & \mathbf{L_o} & & & \end{bmatrix} \tag{4.27}$$

and $\mathbf{L_o}$ is a $m - m_o$ row and m_o column matrix.

We shall call $\mathbf{N_R}$ the reduced matrix of $\mathbf{N}$ and $\mathbf{L}$ its link matrix [Reder, 1988].

Notice that if the rows of $\mathbf{N}$ are independent, i.e. the rank m_o of $\mathbf{N}$ is equal to m, this decomposition is trivial: $\mathbf{N_R}$ is equal to $\mathbf{N}$ and $\mathbf{L}$ is the m dimensional identity matrix.

For the aspartate aminotransferase-malate dehydrogenase system, the first two rows of the scheme matrix $\mathbf{N}$ are independent, the rank m_o of $\mathbf{N}$ is equal to 2, and the following decomposition can be easily worked out either using computer algebra system or by hand:

$$\mathbf{N} = \begin{bmatrix} -1 & 0 \\ 1 & -1 \\ -1 & 0 \\ 1 & 0 \\ 0 & -1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & & 0 \\ 0 & & 1 \\ - & - & - \\ & \mathbf{L_o} & \end{bmatrix} \cdot \begin{bmatrix} -1 & 0 \\ 1 & -1 \end{bmatrix} \tag{4.28}$$

The reduced matrix $\mathbf{N_R}$ is obtained by extracting from $\mathbf{N}$ its m_o rows. Note we have renumbered the matrix $\mathbf{N}$, then its first two rows become linearly independent. The matrix $\mathbf{L_o}$ is constituted from the five last rows of $\mathbf{L}$ as follows:

$$\mathbf{N} = \mathbf{L} \cdot \mathbf{N_R} \tag{4.29}$$

therefore,

$$\mathbf{L} = \mathbf{N} \cdot \mathbf{N_R^{-1}} = \begin{bmatrix} -1 & 0 \\ 1 & -1 \\ -1 & 0 \\ 1 & 0 \\ 0 & -1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} -1 & 0 \\ -1 & -1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -- & -- \\ 1 & 0 \\ -1 & 0 \\ 1 & 1 \\ -1 & -1 \\ -1 & -1 \end{bmatrix} \tag{4.30}$$

Thus,

$$\mathbf{L_o} = \begin{bmatrix} 1 & 0 \\ -1 & 0 \\ 1 & 1 \\ -1 & -1 \\ -1 & -1 \end{bmatrix} \tag{4.31}$$

As the rows of **N** are associated with the components of the metabolite concentration vector **X**, it is natural to decompose also the vector **X** into its first m_o components $\mathbf{X_R}$ and its $m - m_o$ last components $\mathbf{X_R'}$.

$$\mathbf{X} = \begin{bmatrix} \mathbf{X_R} \\ \mathbf{X_R'} \end{bmatrix} = \begin{bmatrix} asp \\ oaa \\ - - - - - \\ \alpha\text{-kg} \\ glu \\ NADH \\ mal \\ \text{NAD}^+ \end{bmatrix} \tag{4.32}$$

The following result is obtained [Reder, 1988].

Every structural conservation relationship is a linear combination of the $\mathrm{m} - \mathrm{m_o}$

independent conservation relationships described by the equality:

$$\frac{d}{dt}(\mathbf{X}_{\mathbf{R}}' - \mathbf{L}_{\mathbf{o}} \cdot \mathbf{X}_{\mathbf{R}}) = 0 \tag{4.33}$$

Therefore, the conservation relationship for aspartate aminotransferase-malate dehydrogenase are as follows;

$$\begin{aligned}
\alpha\text{-kg} - asp &= \alpha\text{-kg}_o - asp_o \\
glu + asp &= glu_o + asp_o \\
NADH - asp - oaa &= NADH_o - asp_o - oaa_o \\
mal + asp + oaa &= mal_o + asp_o + oaa_o \\
\text{NAD}^+ + asp + oaa &= \text{NAD}^+{}_o + asp_o + oaa_o
\end{aligned} \tag{4.34}$$

In other words, the five quantities remain constant.

A similar exercise was performed with the malate dehydrogenase-fumarase and malate dehydrogenase isolated systems, whose concentration relationships are:

1. for the malate dehydrogenase-fumarase coupled system

$$\begin{aligned}
oaa - NADH &= oaa_o - NADH_o \\
Fumarate + mal + NADH &= Fumarate_o + mal_o + NADH_o \\
mal + \text{NAD}^+ + NADH &= mal_o + \text{NAD}^+{}_o + NADH_o
\end{aligned} \tag{4.35}$$

2. for the malate dehydrogenase isolated system

$$\begin{aligned}
oaa - NADH &= oaa_o + NADH_o \\
mal + NADH &= mal_o + NADH_o \\
\text{NAD}^+ + NADH &= \text{NAD}^+{}_o + NADH_o.
\end{aligned} \tag{4.36}$$

We use these metabolite conservation relationship to eliminate unwanted quantities in the system.

4.3 Experimental Procedure

A series of experiments were run *in vitro* to test the suitability of this work for looking at complete enzyme systems. The experiments were performed with the *in vitro* system of fumarase, malate dehydrogenase (MDH) and aspartate aminotransferase (AAT). In the body, this is part of the pathway involved in protein biosynthesis, and converts fumarate and glutamate to aspartate and α-ketoglutarate. For practical experimental purposes we have chosen to run it in the reverse direction.

$$\text{aspartate} + \alpha\text{-ketoglutarate} \xrightleftharpoons[\text{AAT}]{v_1} \text{glutamate} + \text{oxaloacetate}$$

$$\text{oxaloacetate} + \text{NADH} \xrightleftharpoons[\text{MDH}]{v_2} \text{malate} + NAD^+$$

$$\text{malate} \xrightleftharpoons[\text{fumarase}]{v_3} \text{fumarate}$$

We run the reaction in a closed vessel, starting off with just aspartate, α-ketoglutarate and NADH. The progress is followed by UV-spectroscopy to monitor the breakdown of NADH. This data is logged automatically every few seconds by an IBM PC connected to an analogue to digital converter. The experiments have been provided by D.L. Fisher [Fisher, 1990a-b]. We have used a single experiment for every coupled and isolated system.

We have concentrated solely on three enzyme systems of aspartate aminotransferase-malate dehydrogenase, malate dehydrogenase-fumarase and malate dehydrogenase respectively. We make our quasi-steady state assumption that $V = v_1 = v_2$ for aspartate aminotransferase-malate dehydrogenase and $v = v_2 = v_3$ for malate dehydrogenase-fumarase.

4.3.1 Multiple Enzyme Systems

The experimental procedure has been described. For the aspartate aminotransferase malate dehydrogenase system, the initial metabolite concentrations were as follows;

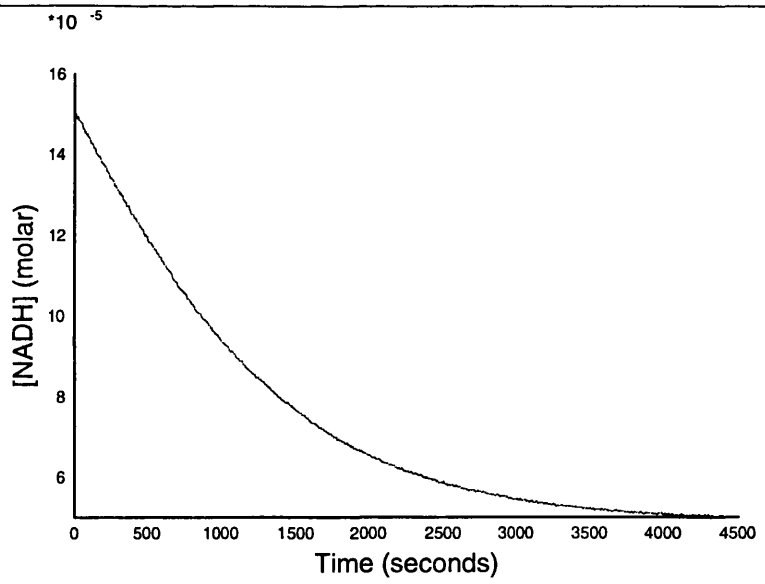


Figure 4-1: Graph of $NADH$ time course data in the aspartate aminotransferase-malate dehydrogenase coupled system. The 732 points were collected spectrophotometrically. The experiment was performed at $30°C$.

Aspartate	=	248.4	mM
α-ketoglutarate	=	0.100	mM
Glutamate	=	0.0	mM
Oxaloacetate	=	0.0	mM
NADH	=	0.150	mM
Malate	=	0.0	mM
NAD^+	=	0.0	mM

and enzyme concentrations were:

Aspartate aminotransferase	=	0.5131	μM
Malate dehydrogenase	=	8.652	μM

Data collection was performed using an analogue to digital converting package for an IBM PC to obtain accurate spectrophotometer output data at $2-10$ second intervals over a period of about 4500 seconds at $30°C$. These data points are shown in figure 4-1.

Our second experiment was for the malate dehydrogenase-fumarase system. It used the following enzyme concentrations:

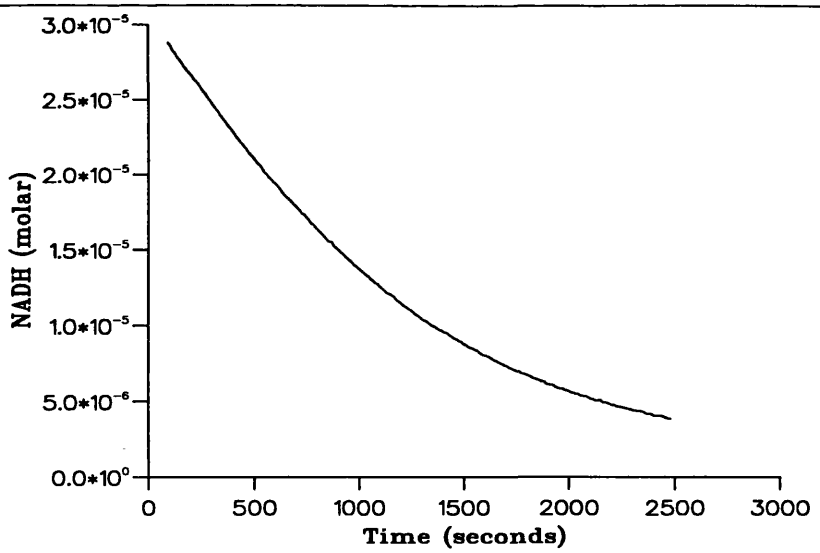


Figure 4-2: Experimental $NADH$ time course data in a malate dehydrogenase-fumarase coupled system.

Fumarase	=	2.99000	μM
Malate dehydrogenase	=	0.02761	μM

Initial metabolite concentrations were

Fumarate	=	0.0	mM
Oxaloacetate	=	5.0000	mM
NADH	=	0.0288	mM
Malate	=	0.0	mM
NAD^+	=	0.0	mM.

The 222 data points were logged automatically using an analogue to digital converter with an IBM PC and shown in figure 4-2.

4.3.2 Single Enzyme System

The experimental procedure is given here for a malate dehydrogenase isolated system. The initial metabolite concentrations were:

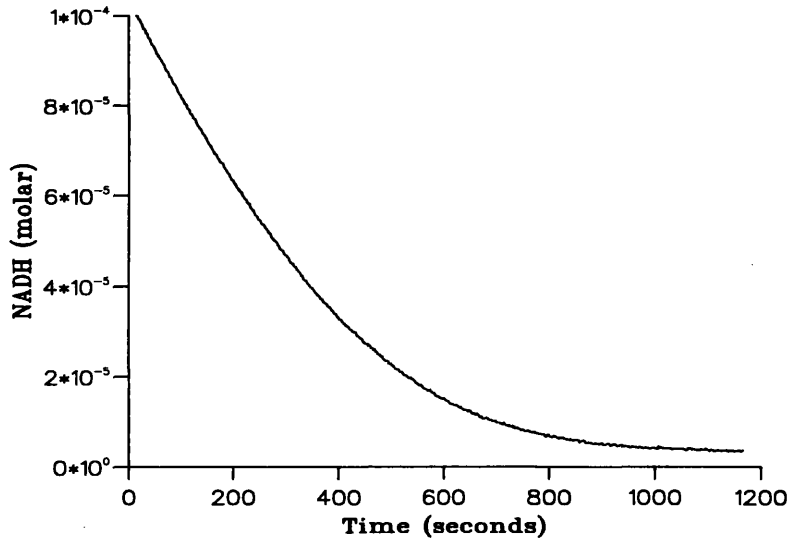


Figure 4-3: Experimental $NADH$ time course data in a malate dehydrogenase isolated system.

Oxaloacetate	=	0.200	mM
NADH	=	0.100	mM
Malate	=	0.0	mM
NAD^+	=	0.0	mM

The enzyme concentration was

$$\text{Malate dehydrogenase} = 0.2173 \quad \mu\text{M}.$$

Data collection was performed using an analogue to digital converting package for an IBM PC to obtain accurate spectrophotometer output data at $2-10$ second intervals over a period about 1165 seconds at $25°C$, and shown in figure 4-3.

4.4 Numerical Simulation

The simulation of the metabolism involves obtaining solutions to rate laws, v_1 and v_2, by integration. In the following section we will perform simulation of the aspartate aminotransferase-malate dehydrogenase coupled system using REDUCE and the numerical analysis package, NAG.

4.4.1 Conventional Simulation of System Behaviour

The objective at this stage was to perform a conventional simulation of the quantitative behaviour of the metabolites in the enzyme system. This would not involve the assumption of the existence of a steady state of the intermediate metabolites.

A NAG routine was used to integrate the rate laws numerically, v_1 and v_2, and simulate changes in metabolite concentrations, based on initial starting concentrations of metabolites and enzymes, and the five concentration equations which define the system. Published values of K_M, K_I and V_{max} were used.

The FORTRAN forms of the enzyme rate laws and their higher derivatives were generated using the GENTRAN package, which produces formatted FORTRAN versions of the symbolic REDUCE formulae and writes them to a template file to produce an executable program when compiled. The result of the integration by this program was the calculation of the concentrations of *asp* and $NADH$ respectively as a function of time. The FORTRAN program used the NAG routines $D02EBF$ and $D02EJY$. $D02EBF$ uses a variable order, variable step method to integrate a system of first order differential equations (in this case, derivatives of product formation with respect to time). $D02EJY$ supplied $D02EBF$ with the Jacobian matrix of partial derivatives $\partial F_i/\partial Y_i$, where F_i is an enzyme activity, v_1 or v_2, and Y_i is a metabolite concentration, *asp* or $NADH$ (Note derivatives enormous if done analytically in REDUCE).

Knowing *asp* and $NADH$ all the metabolite concentrations could then be derived using the following REDUCE code:

```
% Rate law for v1. ( equ6 = num( v - v1 )).
     bas  := groebner({ equ1, equ2, equ3, equ4, equ5, equ6 },
                      { akg,  glu,  oaa,  mal,  nad,  v    }) $
     DY(1) := -part( solve( part( bas, 6 ), v ), 1, 2 ) $
% Rate law for v2, (equ7 = num( v - v2 )).
     bas  := groebner({ equ1, equ2, equ3, equ4, equ5, equ7 },
                      { akg,  glu,  oaa,  mal,  nad,  v    }) $
     DY(2) := -part( solve( part( bas, 6 ), v ), 1, 2 ) $
% Relationship between the metabolites.
```

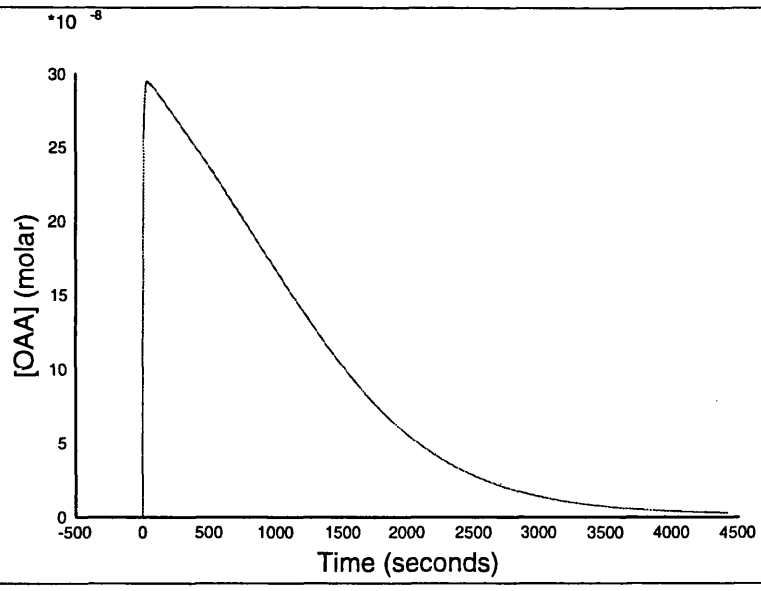


Figure 4-4: Simulated *oaa*, the intermediate metabolite in the aspartate aminotransferase-malate dehydrogenase coupled system.

```
bas := groebner({ equ1, equ2, equ3, equ4, equ5 },
                { akg,  glu,  oaa,  mal,  nad }) $
nad := part( solve( part( bas, 5 ), nad ), 1, 2 ) $
mal := part( solve( part( bas, 4 ), mal ), 1, 2 ) $
oaa := part( solve( part( bas, 3 ), oaa ), 1, 2 ) $
glu := part( solve( part( bas, 2 ), glu ), 1, 2 ) $
akg := part( solve( part( bas, 1 ), akg ), 1, 2 ) $
```

Where **equ1, equ2, equ3, equ4 and equ5** are 4.34, and equation **v1** is 4.7 and **v2** is 4.13. The result of the simulation are displayed graphically in figures 4-4, 4-5 and 4-6. It can be seen that the simulation predicts that the system is well coupled and close to a steady state early in the time course.

4.4.2 Elimination of the Variables in the Steady State

In this section the objective is to derive an equation for the steady state behaviour of the system including only terms in V (the system flux), NADH–the variable being observed experimentally, and the kinetic parameters, the numerical values of which

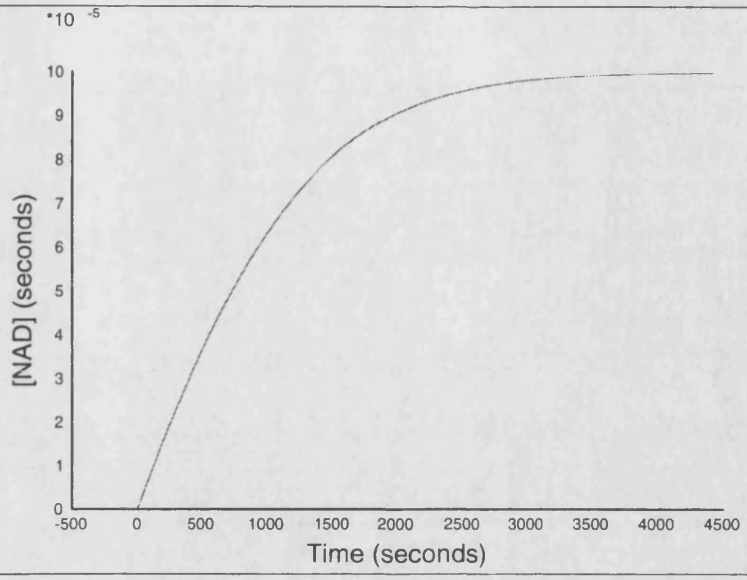


Figure 4-5: Simulated rise of NAD^+, the product in the aspartate aminotransferase-malate dehydrogenase coupled system.

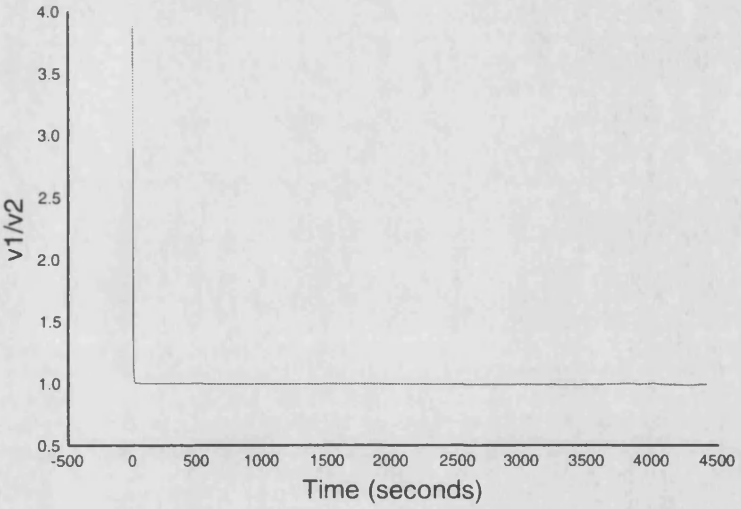


Figure 4-6: Ratio of two enzyme rates. At steady State this ratio is 1.0.

(taken from the literature) could be substituted. This equation is obtained, in practice, by the use of Buchberger's algorithm to calculate a Gröbner basis on the assumption that the two enzymes are coupled *i.e.* they are at quasi-steady state ($V = v_1 = v_2$). With the five metabolite conservation relationships, this gave seven simultaneous polynomial equations. Terms in all metabolites except NADH were eliminated by computation of a lexicographically ordered Gröbner basis (see [Davenport *et al*, 1988] for more theory on algebraic ordering).

This basis took approximately 11 seconds computer time to calculate and yielded a cubic equation in V, the coefficients of which were polynomials of degree 10 in NADH; all of the other metabolite concentration variables were eliminated from this equation. The basis also included equations for the other metabolites, which were linear in V.

The numerical solution of the equation for V is a straightforward matter–the solution of a simple cubic equation. The NAG routine `C02AEF` (NAG, 1988) was employed, which finds all roots of a real polynomial equation. By using this method, rather than an analytical solution for V, the coefficients of V are generated as rational numbers (simplifying the situation), and it is the numerical value of the root for V that is required. The four coefficients for the cubic equation were generated for each time-point by substitution of the respective experimentally observed NADH value. On successful exit from the NAG routine, the real roots to the equation were obtained. This was repeated for all the time-points. In principal REDUCE could find roots analytically, but V, big formulae.

The equation in V has three roots, of which either one or three might be real. In practice, only one of these would be meaningful. With three real roots, which was the situation obtained, only one could be a stable solution–*i.e.* lead to concentrations of all metabolites having positive values (unless the experimental system had multiple steady states which would have been, to say the least, surprising!). It was possible to deduce the correct root by substituting them into the other six parts of the Gröbner basis ideal to obtain values for all the metabolite concentrations. No metabolite concentration could have a negative value and the single meaningful root was obtained by a process of elimination. It is shown in figure 4-7. This root, $V = -d[NADH/dt]$ and it was

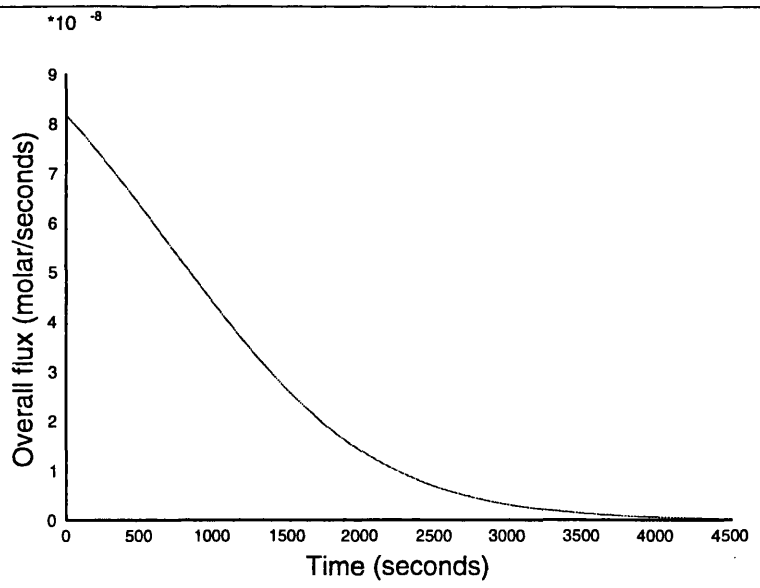


Figure 4-7: The calculated meaningful root of velocity, as a function of experimental $[NADH]$, over the reaction time course in aspartate aminotransferase-malate dehydrogenase coupled system.

integrated by numerical simulation. For this process, the NAG library routine $D02EDF$ was used and the FORTRAN form of the enzyme flux law was generated using the GENTRAN. The result of the simulation is displayed in figure 4-8. The result showed that for most of the time period of the reaction, there is good agreement between calculated and experimentally observed $NADH$ values. The simulation showed that the $V = v_1 = v_2$ assumption was justified.

4.4.3 Simulation of Malate Dehydrogenase-Fumarase coupled system

The system flux equation was obtained using the quasi-steady state assumption, i.e. $v = v_2 = v_3$, with the three metabolite conservation relationships (4.35). These gave five simultaneous, non-linear equations which were solved using Gröbner bases. We obtained a flux equation which was quadratic in v. The solution was obtained using an interfacing routine which substitutes the value of the $NADH$ into the flux equation in REDUCE. This process was done for each data point, and only one of the two roots obtained was meaningful (see figure 4-9). Next the solution was converted into the FORTRAN form using an interfacing routine to perform numerical integration. The NAG routine

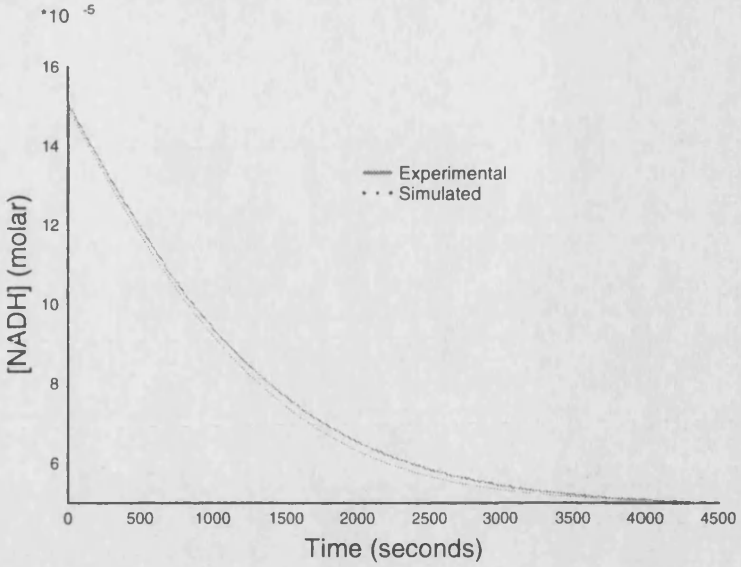


Figure 4-8: Graph of the output from the simulation in the aspartate aminotransferase-malate dehydrogenase coupled system. It shows that there is good agreement between experimental and simulated $[NADH]$ values.

$D02EDF$ was used for numerical integration. The result is shown in figures 4-11, 4-10 and 4-12. As can be seen in figure 4-11, the simulated reaction was faster than was experimentally observed. By fitting the kinetic parameters to the observed data, it is possible to obtain much closer fits (see chapter 5).

4.4.4 Simulation of Malate Dehydrogenase Isolated System

To obtain an equation in terms of v_2 and $NADH$, we eliminated all unwanted quantaties using three conservation relationships, equations 4.36. The flux equation was linear in v_2. We substituted value of $NADH$ into the flux equation for each data point to obtain numerical value of v_2. This process was done in REDUCE, then the value of v_2 was converted in the FORTRAN form using an interfacing routine. Next the NAG routine $D02EDF$ was used to perform the numerical integration. The result is displayed in figure 4-13. Again we see divergence in the reaction rates, and fitting of kinetic parameters would be appropriate.

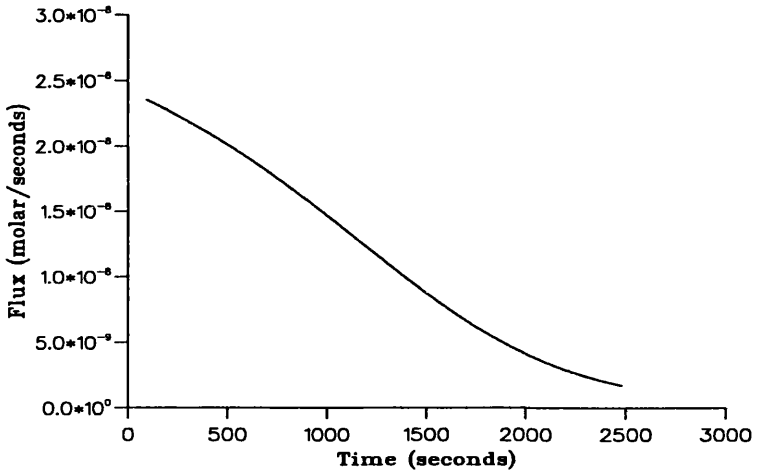


Figure 4-9: The calculated meaningful root of velocity, as a function of experimental $[NADH]$, over the reaction time course in the malate dehydrogenase-fumarase coupled system.

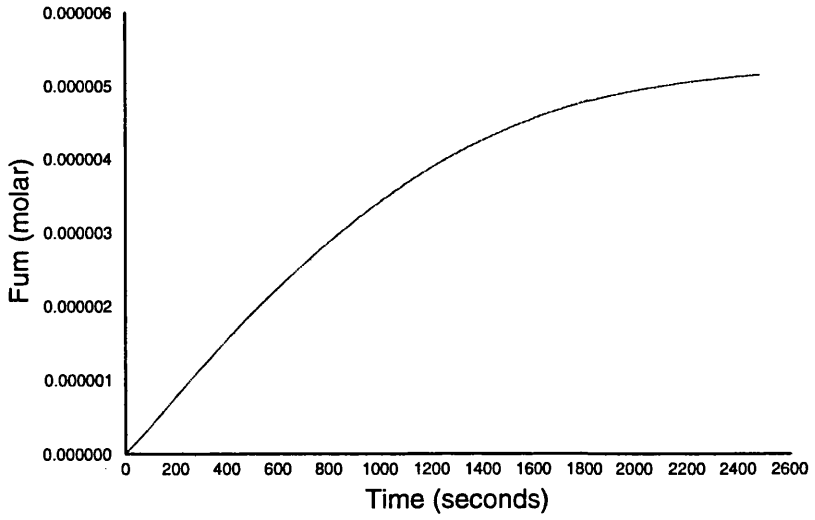


Figure 4-10: Simulated rise of fumarate, the product in the malate dehydrogenase-fumarase coupled system.

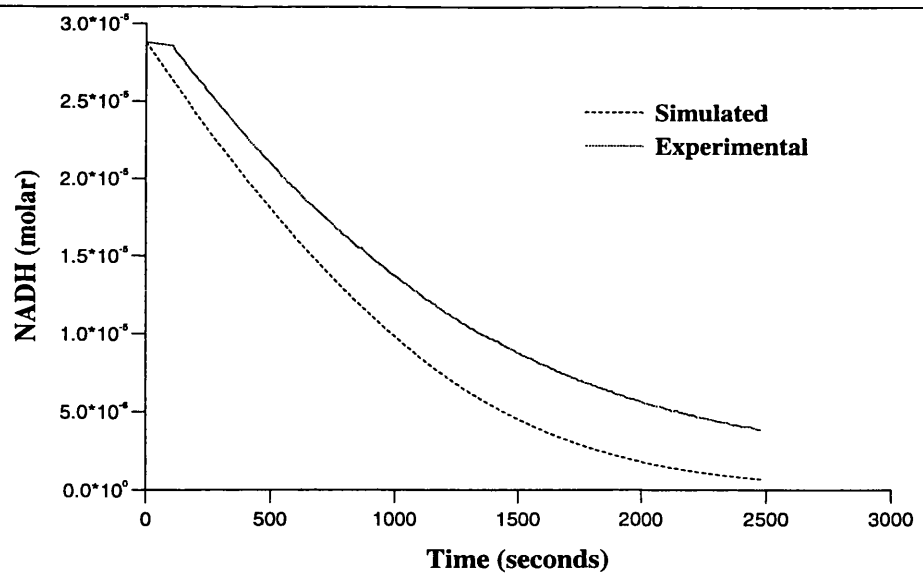


Figure 4-11: Comparison of the simulated and experimentally observed $[NADH]$ time course in the malate dehydrogenase- fumarase system.

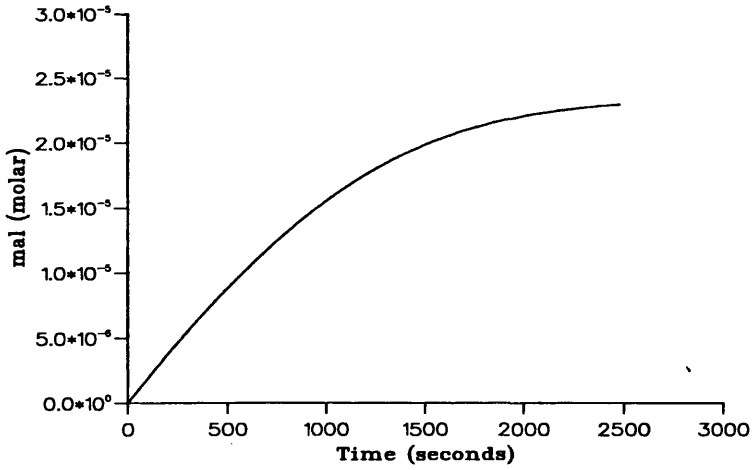


Figure 4-12: Simulated rise of malate, the intermediate metabolite in the malate dehydrogenase-fumarase coupled system.

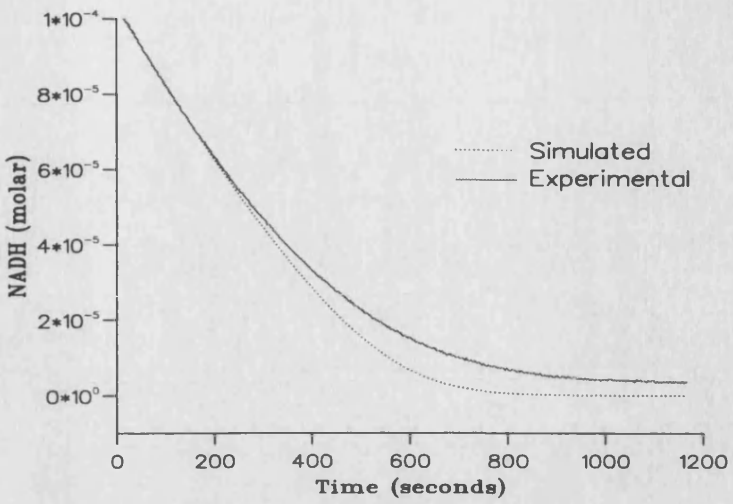


Figure 4-13: Simulated and experimentally observed [NADH] time course in the malate dehydrogenase isolated system.

4.4.5 Summary

A working basis for the solution of the kinetics of enzyme systems–aspartate aminotransferase-malate dehydrogenase, malate dehydrogenase-fumarase and malate dehydrogenase isolated–is presented for the application of computer algebra techniques, such as REDUCE. Both numerical simulation, using rate equations and kinetic parameters reported in the literature, and prediction from the knowledge of behaviour of coupled enzyme systems, suggest that the system (aspartate aminotransferase-malate dehydrogenase coupled system) employed here should be a good model of steady state system.

The simulation performed here showed a similar time-course for $NADH$, the measured variable, to experimental values. Any discrepancy may be due to inapplicability of the various kinetic constants. It has been noted previously [Jenkins and Fonda, 1984] that differences in experimental conditions have led to a variation in the values of kinetic constants for aspartate aminotransferase as determined by different groups of workers. Finally we must allow few the occurence of random experimantal error.

Chapter 5

Parameter Estimation

Determination of enzyme kinetic parameters has traditionally been done by isolating individual enzymes in *vitro* and performing initial rate experiments [Reich *et al*, 1972; Chandler, 1972; Fernley, 1974; Swartz and Brevermann, 1975]. Linearizations of the formulae governing enzyme behaviour allow these data to be plotted, and fitting a straight line yields the kinetic parameters of the enzyme mechanism [Atkins and Nimmo, 1973; Atkins, 1973; Bates and Frieden, 1973a-c; Eisenthal and Cornish-Bowden, 1974; Cornish-Bowden and Eisenthal, 1974 and Waley, 1981].

The nature of experimental error in the determination of initial velocities of enzyme catalysed reactions was investigated [Storer *et al*, 1975]. The quantitative treatment of experimental data requires an adequate consideration of the reliability of the dependent and independent variables. In enzyme kinetics this problem has been appreciated by numerous workers [Wilkinson, 1961; Cleland, 1967 and 1969; Reich, 1970 and Askelof *et al*, 1976].

The analysis of progress curves for enzyme catalysed reactions by non-linear regression has been studied [Duggleby and Morrison, 1977; Duggleby, 1985; Cornish-Bowden and Endrenyi, 1986; Boeker, 1987; Leatherbarrow, 1990; Duggleby, 1991]. A procedure, based on the Gauss-Newton method for non-linear regression, has been developed to obtain enzyme kinetic constants from the analysis of progress curve data. The method of progress curve analysis for enzyme catalysed reactions has been extended to a two substrate, reversible reaction through the use of enzyme catalysed recycling of

	parameters	published value	parameters	published value
AAT	K_M^{asp}	2.0×10^{-3}	K_I^{asp}	2.50×10^{-3}
	$K_M^{\alpha\text{-kg}}$	1.0×10^{-4}	$K_I^{\alpha\text{-kg}}$	6.0×10^{-4}
	K_M^{oaa}	4.0×10^{-5}	K_I^{oaa}	3.6×10^{-5}
	K_M^{glu}	4.0×10^{-3}	K_I^{glu}	6.7×10^{-3}
	V_f^{cat}	322	V_r^{cat}	1342

Table 5.1: The kinetic parameters for aspartate aminotransferase (AAT). They were left in symbolic form in the rate law (v_1) to calculate the overall rate equation. Units for catalytic constants are sec^{-1} and other constants are M. The values for the kinetic parameters have been reported [Velick and Vavra, 1962].

one of the products [Duggleby and Morrison, 1978 and Wilkinson, 1961].

A combination of the jackknife [Efron, 1979] and non-linear regression techniques to measurements of the rate constants for enzyme catalysed reactions have been applied [Cornish-Bowden and Wong, 1978]. In this chapter we have improved a method to fit kinetic parameters to rate laws using experimental data.

5.1 Estimation of Kinetic Parameters for Coupled System

One of the reasons for using computer algebra was to be able to determine the kinetic parameters, the values of which make up the numerical coefficients in the equations described in the previous section. For this an analogous form of the equation (4.7, 4.13 and 4.19) would be required but with coefficients of polynomials in these constants, i.e. without substitution of their values. There are 19 such kinetic parameters needed to define for aspartate aminotransferase-malate dehydrogenase, and 14 for malate dehydrogenase-fumarase coupled systems. These parameters are given in table 5.1, 5.2 and 5.3.

Once the equation is calculated in terms of $\mathbf{V}$, $NADH$ and number of kinetic parameters then we smooth the data points and take the derivative to get $\mathbf{V}$ and $NADH$, the value of each data pair of $\mathbf{V}$ and $NADH$ could be substituted into the equation in order to obtain a set of polynomials in the kinetic constants (since the original equation must belong to the ideal for the ring in which it is calculated, all

	parameters	published value	parameters	published value
MDH	K_M^{oaa}	4.13×10^{-5}	K_I^{oaa}	5.00×10^{-4}
	K_M^{NADH}	1.70×10^{-5}	K_I^{NADH}	5.15×10^{-6}
	K_M^{mal}	4.15×10^{-3}	K_I^{mal}	1.68×10^{-1}
	K_M^{NAD}	3.03×10^{-4}	K_I^{NAD}	1.04×10^{-3}
	V_f^{cat}	1364	V_r^{cat}	220

Table 5.2: The kinetic parameters for malate dehydrogenase (MDH). They were left in symbolic form in the rate law (v_2) to calculate overall rate equation. Units for catalytic constants are sec^{-1} and other constants are M. The values for the kinetic parameters have been reported [Raval and Wolfe, 1962].

	parameters	published value	parameters	published value
FUM	K_M^{mal}	2.5×10^{-5}	V_f^{cat}	900×10^{-4}
	K_M^{fum}	5.0×10^{-6}	V_r^{cat}	800

Table 5.3: The kinetic parameters for fumarase. They were left in symbolic form in the rate law (v_3) to calculate overall rate equation. Units for catalytic constants are sec^{-1} and other constants are M. The values for the kinetic parameters have been reported [Bergmeyer, 1981].

the polynomials generated should be equal to zero). In theory it should be possible to obtain values for the kinetic constants by using a utility routine for minimisation of a function by fitting values to variables describing that function. So we can write down our equation as:

$$\mathbf{V} = g(NADH, k_{i=1}^{19}) \tag{5.1}$$

We can perform a least squares fit on this equation to get an estimate of the kinetic parameters k_i from our experimental data [Bennett, Davenport and Sauro, 1988]. Statistically, it is desirable to find a minimum for the sum of squares, i.e. minimise

$$F = \sum_{i=1}^{n} [f(\mathbf{V_{obs_i}} - g(NADH_i, k_{j=1}^{19}))]^2 \tag{5.2}$$

where V_{obs_i} is experimental V. Calculation of this equation in **V**, NADH and the 19 kinetic constants took hundreds of computer seconds. A different term order was used to calculate Gröbner basis. This used the following format:

```
torder totaldegree;
bas1 := groebner({ equ1, equ2, equ3, equ4, equ5, equ6, equ7 },
                 { asp, akg, glu, oaa, mal, nad, V }) ;
torder invlex ;

bas := groebner( bas1, { asp, akg, glu, oaa, mal, nad, V });
```

where **equ1, equ2, equ3, equ4** and **equ5** are conservation relationships, i.e. they are 4.34, and **equ6** and **equ7** are rate equations, i.e. (**equ6** = V-v_1 and **equ7** = V-v_2). Second REDUCE format was

```
on groebopt;
bas1 := ({ equ1, equ2, equ3, equ4, equ5, equ6, equ7 },
                  { asp, akg, glu, oaa, mal, nad, V }) ;
%
bas := glexconvert( bas1 , gvarslast , newvars = {V} ) ;
equation:= part( bas , 1 ) ;
```

Here **glexconvert** converts an arbitrary Gröbner basis into a lexicographical order one, **gvarslast** is list of the variables which are stored as REDUCE list in the **bas1** and **newvar** defines the new variable sequence.

Above calculations might be faster, in order to obtain the correct equation from this, it would be necessary to revert to lexicographical ordering on passing in the list of metabolite concentrations to be eliminated. This approach was unsuccessful: an attempt to derive an equation containing 19 symbolically defined constants had not terminated after several hours.

A different method was used to solve seven equations simultaneously. Five conservation equations (4.34) were linear with respect to each metabolite concentration variable. These five equations were solved simultaneously using REDUCE linear algebra facilities and each metabolite concentration variable was obtained in term of $NADH$ and *oaa*. These values were substituted into last two equations ($num(V - v_1)$, $num(V - v_2)$) and two polynomials were found:

$$\begin{aligned} equ6 &= f_a(V, NADH, oaa, k_i) \\ equ7 &= g_a(V, NADH, oaa, k_i) \end{aligned} \tag{5.3}$$

These two equations were solved with respect to *oaa*, using **resultant**, and the variable *oaa* eliminated.

Resultant computes the resultant of the two given polynomials with respect to the given variable. The result can be identified as the determinant of a Sylvester matrix, but can often also be thought of informally as the result obtained when the given variable is eliminated between the two input polynomials. If the two input polynomials have a non-trivial **gcd**, their resultant vanishes (see [Davenport, Siret and Tournier, 1988] for more information).

The calculation of the resultant of the given polynomials, took about 10 minutes on a Sun-4/260, and yielded a polynomial in terms of **V**, NADH and the 19 kinetic parameters, which takes nearly 3204 lines (nearly 120 A4 pages) to print out. The polynomial was rather large, involving coefficients of the order of 10^{52}, and the derivative function with respect to each kinetic parameter was 27560 lines. Evaluating these large

expressions in FORTRAN is prone to rounding error.

Thus a simpler system was necessary to test the commensurability of minimisation of function techniques to estimating kinetic constants. This problem will be discussed in the next sections. Another important problem were the selection of the minimisation and integration routines, because some minimisation routines were very sensitive to the function to be used and some routines need first and second derivatives of the function to be evaluated. Before handling the simpler system we will give statistical error estimation technique called *Bootstrap.*

5.2 Statistical Analysis: Bootstrap

When a minimum for a function of the form

$$F = \sum_{i=1}^{n} [f(\mathbf{V_{obs_i}} - g(NADH_i, k_{j=1}^{19}))]^2 \tag{5.4}$$

has been found, each $[f(\mathbf{V_{obs_i}} - g(NADH_i, k_{j=1}^{19}))]^2$ will yield a small residual value using the fitted estimates. There will therefore be i such residuals for any set of data. By storing these as they are generated, they can be used later to estimate standard deviations for the fitted parameters. This was achieved by a technique called bootstrap [Efron, 1979]. If the set of fitted estimates for the parameters are denoted $\hat{k}$, a new set of estimates $\hat{\hat{k}}$ can be obtained by taking, in turn, one of the i residuals at random and calculating, using one of the values of $\mathbf{V_{obs}}$, a value of $\mathbf{V_{theo}}$, that would give this residual using the estimates $\hat{k}$. This was done for all of the data values $\mathbf{V_{obs}}$, to obtain i new values for $\mathbf{V_{theo}}$ These were then used as data, substituting for $\mathbf{V_{obs}}$ in a second round of minimisation to calculate new estimates $\hat{\hat{k}}$. after repeating this many times, the statistical parameters for $\hat{\hat{k}}$ were calculated. The distribution of this is assumed to be similar to that for $\hat{k}$ around k.

5.2.1 Malate Dehydrogenase System

In the previous section, it was stated that it was not easy to evaluate a large expressions in FORTRAN form. So in this section, we consider a simpler system, the

malate dehydrogenase isolated system:

$$\text{oaa} + \text{NADH} \underset{\text{MDH}}{\overset{v_2}{\rightleftharpoons}} \text{mal} + \text{NAD}$$

The polynomial function in v_2, NADH and the 10 kinetic parameters took 408 ms to calculate. It was linear with respect to v_2, i.e. of the form

$$\alpha \times v_2 + \beta = 0 \tag{5.5}$$

where α and β are terms in the kinetic constants and in NADH. The following section describes fitting of the kinetic constants to this function. Several NAG routines were used to find the minimum of the function. To simplify matters we only tried to estimate K_M^{mal}, giving the remaining parameters their published numerical values.

The NAG routines $E04ABF$ and $E04BBF$ were employed for minimisation of the function in a given finite interval. The methods ($E04ABF$ is based on quadratic and $E04BBF$ is based on cubic interpolation) are intended for functions which have a continuous first derivative, although they work usually even if the derivative has occasional discontinuities.

The function was calculated in REDUCE, and a FORTRAN expression was generated using GENTRAN and incorporated into subroutine FUNCT. Data was used here which was obtained by solving the rate law v_2 as a initial value problem such that

$$\begin{aligned} &v_2 = -d[NADH]/dt = f(NADH, t, k_i; i = 1, 10) \\ &NADH(t_o) = 1.00 \times 10^{-4} M \ \ and \ \ t_o \leq t \leq t_{max}. \end{aligned} \tag{5.6}$$

We used the published numerical values of kinetic parameters k_i and recorded 564 data point. The data was for v_2 against NADH concentration (after first minimisation of the function

$$F = \sum_{i=1}^{n} [f(v_{2_{obs_i}} - g(NADH_i, k_M^{mal}))]^2). \tag{5.7}$$

We recorded the difference between the numerical and experimental values (experimental value means that simulated value) as a residual error. We then used standard least-squares minimisation to obtain estimates of kinetic parameter K_M^{mal}. The procedure used is as follows:

Routine	Parameter	Actual val.	Experimental val.	Sum of squ.
$E04ABF$	K_M^{mal}	4.15×10^{-3}	$3.604 \pm 0.546 \times 10^{-3}$	1.478×10^{-11}
$E0ABF$	K_M^{mal}	4.15×10^{-3}	$3.750 \pm 0.300 \times 10^{-3}$	1.470×10^{-11}

Table 5.4: Result of parameter estimation using simulated data. Unit for sum of squares are $(\mathbf{Molar}/Second)^2$.

1. Initial estimate for the parameter K_M^{mal} is provided.

2. The function

$$f = v_{2_{obs_i}} - v_{2_{theor_i}} \tag{5.8}$$

   is computed.

3. The optimisation technique is used to find a smaller value of

$$F = \sum f^2 \tag{5.9}$$

4. The process is iterated. It is terminated either after a predeterminated number of iterations, or when **F** becomes approximately constant over several iterations.

The results obtained are displayed in table 5.4.

Quasi-Newton Methods

Quasi-Newton methods approximate the Hessian $G(k^\eta)$ by a matrix B^η which is modified at each iteration to include information obtained about the curvature of F along the latest search direction.

Let s_η be the step taken from k_η, and consider expanding the gradient function (gr) about k_η in a Taylor series along s_η:

$$gr(k_\eta + s_\eta) \approx gr_\eta + G_\eta s_\eta + \cdots \tag{5.10}$$

The curvature of F along s_η is given by $s_\eta^T G_\eta s_\eta$, which can be approximated using only first-order information:

$$s_\eta^T G_\eta s_\eta = (gr(k_\eta + s_\eta) - gr_\eta)^T s_\eta \tag{5.11}$$

At the beginning of the η-*th* iteration of a quasi-Newton method, an approximate Hessian matrix B_η is available, which is intended to reflect to curvature information already accumulated. If B_η is taken as the Hessian matrix of a quadratic model function, the search direction p_η is the solution of a linear system:

$$B_\eta p_\eta = -gr_\eta \tag{5.12}$$

The initial Hessian approximation B_o is usually taken as the identity matrix if no additional information is available. With this choice, the first iteration of a quasi-newton method is equivalent to an iteration of the steepest-descant method.

After $k_{\eta+1}$ has been computed, a new Hessian approximation $B_{\eta+1}$ is obtained by updating B_η to take account of the newly-acquired curvature information. An update formula is a definition of $B_{\eta+1}$ of the form

$$B_{\eta+1} = B_\eta + U_\eta, \tag{5.13}$$

where U_η is the update matrix. Let the vector s_η denote the change in k during the η-*th* iteration ($s_\eta \equiv k_{\eta+1} - k_\eta = \alpha_\eta p_\eta$), and let y_η denote the change in gradient ($y_\eta = gr_{\eta+1} - gr_\eta$). The standard condition required of the updated Hessian approximation is that it should approximate the curvature of F along s_η. $B_{\eta+1}$ is thus required to satisfy the so-called *quasi-Newton condition*

$$B_{\eta+1} s_\eta = y_\eta \tag{5.14}$$

During a single iteration, new information is obtained about the second-order behaviour of F along one direction; thus, we would expect $B_{\eta+1}$ to differ from B_η by a matrix of low rank. In fact, the quasi-Newton condition can be satisfied by adding a rank-one matrix to B_η. Assume that

$$B_{\eta+1} = B_\eta + uv^T, \tag{5.15}$$

for some vector u and v. From the 5.14

$$B_{\eta+1}s_\eta = (B_\eta + uv^T)s_\eta = y_\eta, \tag{5.16}$$

and therefore u must be in the direction $y_\eta - B_\eta s_\eta$. We assume that y_η is not equal to $B_\eta s_\eta$. For any vector v such that $v^T s_\eta$ is non-zero, the vector u is given by $(1/v^T s_\eta)(y_\eta - B_\eta s_\eta)$, and $B_{\eta+1}$ is defined as

$$B_{\eta+1} = B_\eta + \frac{1}{v^T s_\eta}(y_\eta - B_\eta s_\eta)v^T. \tag{5.17}$$

Although not as robust as Newton type methods, quasi-Newton methods can be more efficient because $G(k^\eta)$ is not computed, or approximated by finite-differences. The quasi-Newton methods minimise a quadratic function in η iterations (see for more information [Gill, Murray and Wright, 1981]).

We applied the quasi-Newton methods to minimise F. For this purpose, the NAG routine $E04JAF$ was employed for finding a minimum of a function, subject to fixed upper and lower bounds on the variables, using function values only. It is intended for functions which are continuous and have continuous first and second derivatives. This routine does not use derivatives of the function.

The function was calculated in REDUCE [Hearn, 1987] and a FORTRAN version was generated using GENTRAN [Gates, 1987]. It was incorporated into subroutine FUNCT1 which was called upon by $E04JAF$ each time it adjusted the variables to test for a minimum. The ability to specify lower bounds meant that non-negativity constraints could be placed on the kinetic constants.

The function for minimisation was the square of a function containing terms in v_2 and NADH which, if they were eliminated by substitution of a pair of experimental values, would yield an equation containing the kinetic parameter which were the variables for minimisation. To test whether reasonable values would result, data was generated

by simulation, using the reported numerical values of the kinetic constants. The specification for $E04JAF$ suggested that the problem is badly scaled, i.e. error exit (IFAIL = 5) was equal to five. That means there was some doubt about whether the point kinetic constants found by $E04JAF$ is a minimum. The degree of confidence in the result decreases as IFAIL increases.

It was thought best to use a different routine. We changed the NAG routine for the NAG routine $E04KAF$, and employed it to minimise a function subject to fixed upper and lower bounds on the variables, where first derivatives were supplied. These derivatives were calculated in REDUCE, and FORTRAN versions were generated using GENTRAN. They are present in subroutine FUNCT2 which was called upon by $E04KAF$ each time. This routine was unsuccessful, i.e. error exit (IFAIL = 5) was equal to five.

We usually avoided using the NAG routines which needs first and second derivatives of the given function to be evaluated (because expressions of the derivatives are so large). Evaluating these large expressions in FORTRAN is prone to rounding error. We used the NAG least squares specialised routine $E04FDF$ to estimate just one kinetic parameter, K_M^{mal}. This routine does not require first and second derivatives and works on a least squares basis, automatically generating

$$F = \sum_{i=1}^{n} [f_i(v_{2_{obs_i}} - g(NADH_i, k_M^{mal}))]^2 \tag{5.18}$$

from f_i; the f_i are referred to as 'residuals'. The user supplies a subroutine LSFUN1 to evaluate functions f_i at any point k_M^{mal} for the main routine $E04FDF$. From a starting point k_M^{mal} supplied by the user, a sequence of points is generated which is intended to coverge to a local minimum of the sum of squares. These points are generated using estimates of the curvature of F.

The expression for the f_i was generated in FORTRAN by GENTRAN. The results were displayed in table 5.5. Error exit IFAIL was 0 when the process was completed.

Finally, we used the NAG routine $E04FCF$ which is a comprehensive algorithm for finding an unconstrained minimum of a sum of squares of m nonlinear functions in n

Routine	Parameter	Actual val.	Experimental val.	Sum of squ.
$E04JAF$	K_M^{mal}	4.15×10^{-3}	$3.856 \pm 0.244 \times 10^{-3}$	1.462×10^{-11}
$E04KAF$	K_M^{mal}	4.15×10^{-3}	$4.000 \pm 0.023 \times 10^{-3}$	1.480×10^{-11}
$E04FDF$	K_M^{mal}	4.15×10^{-3}	$3.980 \pm 0.019 \times 10^{-3}$	1.455×10^{-11}
$E04FCF$	K_M^{mal}	4.15×10^{-3}	$3.980 \pm 0.019 \times 10^{-3}$	1.455×10^{-11}

Table 5.5: Result of parameter estimation using simulated data. Unit for sum of squares are $(Molar/Second)^2$.

variables. The result is displayed in table 5.5. Using the NAG routine $E04FDF$ and $E04FCF$, we obtained exactly same results, so in generally we use $E04FDF$ in future, because it is simpler to use.

Simplex Methods

The NAG routine $E04CCF$ could be use to estimate kinetic parameters, but the routine cannot be used unless the function and its parameters are scaled. The routine finds an approximation to a minimum of a function of n variables. The method is iterative. A simplex of $n+1$ points is set up in the m dimensional space of the variables under the assumption that the problem has been scaled so that the values of the independent variables at the minimum are of order unity. The starting point provided by the user is the first vertex of the simplex, the remaining n vertices are generated by the routine [Parkinson and Hutchinson, 1972]. The vertex of the simplex with the largest function value is reflected in the centre of gravity of the remaining vertices and the function value at this new point is compared with the remaining function values. Depending on the outcome of this test the new point is accepted or rejected, a further expansion move may be made, or a contraction may be carried out [Nelder and Mead, 1965]. When no further progress can be made the sides of the simplex are reduced in length and the method is repeated.

The NAG routine $E04CCF$ was employed for minimisation of the function. The function was calculated in REDUCE and a FORTRAN version was generated using GENTRAN. It was incorporated into subroutine $FUNCT$, which was called by $E04CCF$ each time, and the results are given in table 5.6. Before calculating the

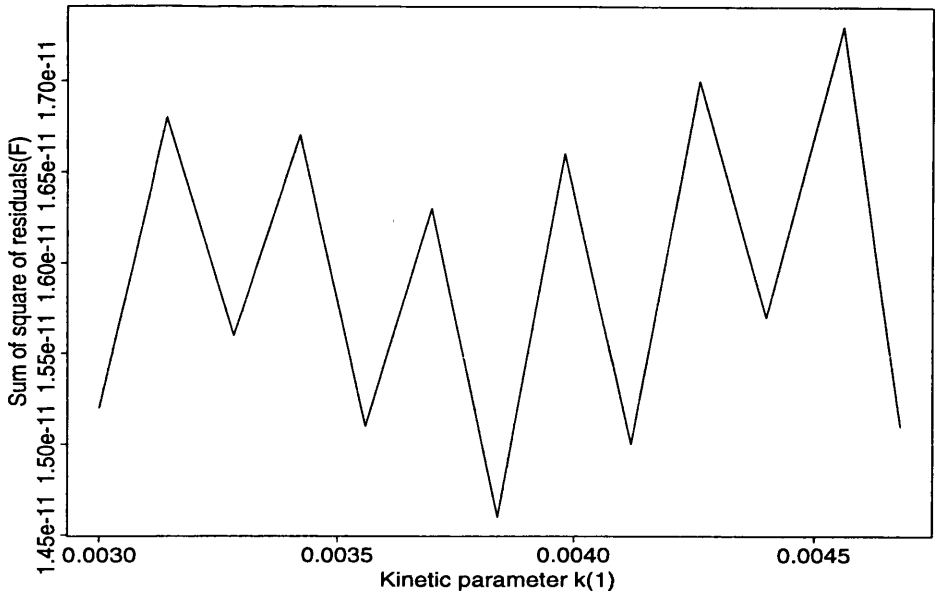


Figure 5-1: A graph of the F at the neighbourhood of the minimum point. k(1) is the kinetic parameter K_M^{mal}. As can be seen from the graph, the F is not smooth because of the computational effect. We have pointed out in the following section the GENTRAN technique has been responsible for introducing errors, which was used to generate FORTRAN expression to evaluate the f.

function in REDUCE, the kinetic parameter K_M^{mal} was scaled so that it had order unity.

The simplex method does not use the gradient vector of the function. When the quasi-Newton method is used, the gradient vector of the function is estimated internally by the NAG routine and used for finding the minimum value of the function. If the function is not smooth at the minimum, then this method can fail because the value of the gradient vectors of the function at the neighbourhood of the minimum are not defined. If we look at the graph of the function at the neighbourhood of the minimum of the function we can see the graph is not smooth (see figure 5-1). This may be just a rounding effect. We have used the NAG least square specialised routines $E04FDF$ and $E04FCF$, and the Simplex method, $E04CCF$, to minimise F. An advantage to use the NAG least squares specialised routine $E04FDF$ is that it takes less computer time and give better results rather then the NAG routine $E04CCF$ (simplex method).

Routine	Parameter	Actual val.	Experimental val.	Sum of squ.
$E04CCF$	K_M^{mal}	4.15×10^{-3}	$3.860 \pm 0.290 \times 10^{-3}$	1.460×10^{-11}

Table 5.6: Result of parameter estimation using simulated data.

Use of the Interfacing Technique

Before scaling the kinetic parameters in given functions, we have evaluated our interfacing technique to the system described in the section 3.3.2. To simplify matters we only tried to estimate k_1 that is the individual kinetic parameter of the system, giving the remaining k_i's their published values.

Our first experiment looked at the sum of the squares of the residuals at the end of the first step of the minimisation process (i.e with our initial guess at k_1). We compared three sets of results:

1. Using a conventional FORTRAN evaluation of the rate law, as produced by GENTRAN.

2. Using REDUCE to evaluate the rate law numerator and denominator exactly, reading them into REDUCE and then carrying out a double precision division.

3. Using REDUCE to evaluate the rate law as a rational number, and then using on `rounded ; precision 15` to express it in REDUCE as a 15 digit floating point number before reading into FORTRAN.

The results are shown in table 5.7. We see that the results using the combined FORTRAN-REDUCE system are different by a factor of more than 2. There is even a slight variation when the last division is carried out in FORTRAN rather than REDUCE.

Our second experiment then compared the performance of the first and third of the above methods in fitting k_1 using just the first 20 experimental data points. The results are shown in table 5.8. We see that the new technique takes nearly 10 times as long to run. However we get an error estimate suggesting that the GENTRAN technique has been responsible for introducing errors.

	(a)	(b)	(c)
Parameter k_1	1.0876	1.095226	1.095223
Sum of sq.	1.260147×10^{-12}	2.813589×10^{-12}	2.813631×10^{-12}

Table 5.7: Sum of squares after first step of minimisation. (a) using GENTRAN to generate all rate law expressions for evaluation in FORTRAN; (b) evaluating numerator and denominator exactly in REDUCE, reading them into FORTRAN and then carrying out a division; (c) evaluating rate law in REDUCE and then using `on rounded ; precision 15` to evaluate it in REDUCE as a floating point number before reading into FORTRAN

	(a)	(b)
User time (seconds)	34.5	210.0
Sys time (seconds)	1.8	141.2
Total	36.3	351.2
k_1 (sec^{-1})	1.142	1.213
Sum of Squares	3.754×10^{-12}	4.662×10^{-12}

Table 5.8: Determination of k_1 by fitting to 20 data points. (a) using GENTRAN to generate all rate law expressions for evaluation in FORTRAN; (b) evaluating the rate law in REDUCE and then using `on rounded ; precision 15` to evaluate it in REDUCE as a floating point number before reading into FORTRAN

5.2.2 Scaling Kinetic Parameters

Scaling often has a significant influence on the performance of optimisation methods. Since convergence tolerances and other criteria are necessarily based on an implicit definition of 'small' and 'large', problems with unusual or unbalanced scaling may cause difficulties for some algorithms.

One method of scaling is to transform the variables from their original representation, which may reflect the physical nature of the problem, to variables that have certain desirable properties in terms of optimisation. We know from the previous section, the NAG minimisation routines suggested that the problem is scaled so that at the solution the value of

$$F = \sum_{i=1}^{n} [f(v_{2i}, NADH_i, k_{i=1}^{10})]^2 \tag{5.19}$$

and the corresponding values of $k_{i=1}^{10}$ are each in the range $(-1, +1)$, and so that at points a unit distance away from the solution, F is approximately a unit value greater than at the minimum.

We used the following scaling format;

$$ksca(j) = k(j) \times \sigma(j) + \lambda(j) \tag{5.20}$$

where ksca(j) are the variables output into FORTRAN and used for minimisation, and $\sigma(j)$ and $\lambda(j)$ indicate the constants needed to bring the value of that kinetic constant into the range of the $(-1, +1)$. If the function,

$$F = \sum_{i=1}^{n} [f(v_{2i}, NADH_i, k_{i=1}^{10})]^2 \tag{5.21}$$

has a minimum at the $k(j)$, then $ksca(j)$ should be equal to zero, and the first derivative of the function with respect to $ksca(j)$ should be -1 or $+$, i.e. $\partial F / \partial ksca(j) = \pm 1$

Therefore we have the following linear equations.

$$\begin{aligned} k(j) \times \sigma(j) + \lambda(j) &= 0 \\ \partial F / \partial ksca(j) \pm 1 &= 0 \end{aligned} \tag{5.22}$$

The above equations were solved simultaneously. The values of $\sigma(j)$ and $\lambda(j)$ obtained were substituted into the algebraic expressions, i.e, in

$$ksca(j) = k(j) \times \sigma(j) + \lambda(j) \tag{5.23}$$

Thus

$$k(j) = (ksca(j)/\sigma(j)) - (\lambda(j)/\sigma(j)) \tag{5.24}$$

Estimates for the values of the variables have to be provided for the minimisation routine. Ideally the true values would be found even when the estimates were quite inaccurate.

Minimisation was applied to the square of the theoretically identical function

$$G = \sum_{i=1}^{n}[g(v_{2theo_i} - v_{2exp_i})]^2 \tag{5.25}$$

because the square of a function such as

$$F = \sum_{i=1}^{n}[f(v_{2i}, NADH_i, k_{i=1}^{10})]^2 \tag{5.26}$$

has a more complicated expression, where v_{2theo} was the theoretical value for v_2, calculated from polynomial 5.5 as

$$v_{2theo} = -\beta/\alpha \tag{5.27}$$

and v_{2exp} was the experimental value for v_2 at that time point. The function

$$FSUM = \sum_{i=1}^{n}[g(v_{2theo_i} - v_{2exp_i})]^2 \tag{5.28}$$

was minimised for each of the 564 sets of simulated data, to yield estimates of $ksca(j)$. The results are displayed in tables 5.9 and 5.10. All kinetic parameters were found much closer to the known values. We have got better results using the NAG least square specialised routines $E04FDF$ and $E04FCF$ which were suggested by Prof. S.

Parameters	Published	Experimental(a)	Experimental(b)
K_M^{oaa}	4.13×10^{-5}	$4.1298 \pm 0.00025 \times 10^{-5}$	$4.1333 \pm 0.00844 \times 10^{-5}$
K_M^{NADH}	1.70×10^{-5}	$1.7015 \pm 0.00006 \times 10^{-5}$	$1.6921 \pm 0.00022 \times 10^{-5}$
K_M^{mal}	4.15×10^{-3}	$4.1588 \pm 0.01280 \times 10^{-3}$	$4.1500 \pm 0.00015 \times 10^{-3}$
K_M^{NAD}	3.03×10^{-4}	$3.0137 \pm 0.01807 \times 10^{-4}$	$3.0310 \pm 0.00028 \times 10^{-4}$
K_I^{oaa}	5.00×10^{-4}	$5.0004 \pm 0.00035 \times 10^{-4}$	$5.0000 \pm 0.00003 \times 10^{-4}$
K_I^{NADH}	5.15×10^{-6}	$5.1585 \pm 0.00105 \times 10^{-6}$	$5.1501 \pm 0.00041 \times 10^{-6}$
K_I^{mal}	1.68×10^{-1}	$1.6123 \pm 0.06254 \times 10^{-1}$	$1.6713 \pm 0.02349 \times 10^{-1}$
K_I^{NAD}	1.04×10^{-3}	$1.0402 \pm 0.00017 \times 10^{-3}$	$1.0410 \pm 0.00072 \times 10^{-3}$
V_f^{cat}	1364	$1.3639 \pm 0.00006 \times 10^{3}$	$1.3620 \pm 0.00045 \times 10^{3}$
V_r^{cat}	220	$2.2059 \pm 0.01254 \times 10^{2}$	$2.2002 \pm 0.00017 \times 10^{2}$
Sum of Squ.		3.2824×10^{-15}	2.7921×10^{-16}

Table 5.9: Fitting of the kinetic parameters to simulated data for MDH isolated system. Parameter fitting was done by bootstrap method. (a) using the quasi-Newton method (NAG routine $E04JAF$). (b) using the simplex method (NAG routine $E04CCF$). Units for sum of squares are $(\mathbf{Molar}/Second)^2$.

Hammarling.

5.2.3 Selection of the Integration Routines

Our overall rate law relates **V** to $NADH$, i.e. we have expressed the rate law as a first order differential equation in the one measurable quantity, $NADH$.

$$DY = -\frac{d[NADH]}{dt} = f(NADH, k_i) \tag{5.29}$$

where k_i's are the enzyme kinetic parameters. We have experimental data for $[NADH]$ against time for the reaction systems described in the previous section. We numerically integrate the rate law and fit it directly to the experimental data. Note that the complexity of the rate law makes analytical differentiation impossible. At this stage we use the numerical analysis library's NAG routines to perform the numerical integration. Several NAG routines were used to integrate the rate law.

The main computational effort in applying the several routines are the evaluation of DY. There is a subroutine, **FCN**, which evaluates the value of DY for each iteration, and is used by main integration routines.

Our experiment compared the performance of the one step integration using three

Parameters	Published	Experimental(a)	Experimental(b)
K_M^{oaa}	4.13×10^{-5}	$4.1321 \pm 0.00336 \times 10^{-5}$	$4.1321 \pm 0.00336 \times 10^{-5}$
K_M^{NADH}	1.70×10^{-5}	$1.6340 \pm 0.00031 \times 10^{-5}$	$1.6340 \pm 0.00027 \times 10^{-5}$
K_M^{mal}	4.15×10^{-3}	$4.1500 \pm 0.00027 \times 10^{-3}$	$4.1500 \pm 0.00027 \times 10^{-3}$
K_M^{NAD}	3.03×10^{-4}	$3.0301 \pm 0.00017 \times 10^{-4}$	$3.0301 \pm 0.00017 \times 10^{-4}$
K_I^{oaa}	5.00×10^{-4}	$5.0000 \pm 0.00001 \times 10^{-4}$	$5.0000 \pm 0.00001 \times 10^{-4}$
K_I^{NADH}	5.15×10^{-6}	$5.1501 \pm 0.00004 \times 10^{-6}$	$5.1501 \pm 0.00004 \times 10^{-6}$
K_I^{mal}	1.68×10^{-1}	$1.6801 \pm 0.00028 \times 10^{-1}$	$1.6801 \pm 0.00028 \times 10^{-1}$
K_I^{NAD}	1.04×10^{-3}	$1.0402 \pm 0.00017 \times 10^{-3}$	$1.0402 \pm 0.00017 \times 10^{-3}$
V_f^{cat}	1364	$1.3639 \pm 0.00004 \times 10^{3}$	$1.3639 \pm 0.00004 \times 10^{3}$
V_r^{cat}	220	$2.2005 \pm 0.00013 \times 10^{2}$	$2.2005 \pm 0.00013 \times 10^{2}$
Sum of Squ.		2.5560×10^{-16}	2.5560×10^{-16}

Table 5.10: Fitting of the kinetic parameters to simulated data for MDH isolated system. Parameter fitting was done by the bootstrap method (a) using the NAG routine $E04FCF$, (b) using the NAG routine $E04FDF$. As can be seen in the table, the results using the NAG routines $E04FCF$ and $E04FDF$ are same. Units for sum of squares are $(Molar/Second)^2$.

numerical integration methods [NAG, 1988]:

1. $E02EAF$—variable-order, variable-step method implementing the backward differentiation formulae—integrates system of first-order ordinary differential equations over an interval with suitable initial condition.
2. $E02BAF$—Runge-Kutta-Merson Method—integrates system of first-order ordinary differential equations over an interval with suitable initial condition.
3. $E02CAF$—variable-order, variable-step Adams method—integrates system of first-order ordinary differential equations over an interval with suitable initial condition.

The results are shown in table 5.11. We see that the use of the NAG routine $D02EAF$ gives more advantages then the others, and it takes less computer time to integrate the given differential equations.

5.2.4 Fitting the Parameters to Experimental data

The logged spectrophotometric data is consist of ordered pairs of time and NADH concentration, i.e. the data is for NADH concentration against time. The rate laws we

	D02EAF	*D02BAF*	*D02CAF*
FCN was called (times)	6349	11191	8058
User time (seconds)	11.260	16.890	12.350
System time (seconds)	5.420	9.40	6.500
Total time	16.680	26.310	18.850

Table 5.11: The subroutine *FCN* is used less by *D02EAF* rather then others routines, and solution of the rate law takes less time using the NAG routine *D02EAF*.

are fitting relate d[NADH]$/dt$ to [NADH]. We need to integrate the rate laws to fit to the data. This is much more reliable but analytical integration of the rate laws is not in general possible. However, we used numerical integration instead. When fitting to integrated equations, v_1 and v_2, it is important not to perform a complete numerical integration over the range and minimise residuals. Such an approach is highly sensitive to errors in the initial conditions.

This is essentially an overconstrained multi-point integration problem. At this stage we resort to numerical techniques and use an established technique to obtain estimates [Bock 1981; Bayram, Bennett and Dewar, 1991]. We use the NAG FORTRAN subroutine library to provide numerical integration and minimisation routines [NAG, 1988]. In summary we provide estimates of k_i, integrate numerically from one experimental data point to the next, and record the difference between the numerical and experimental values as a residual error. We do this for all 732 data points, yielding 731 residual errors. We then use standard least-squares minimisation to obtain better estimates of k_i that will reduce these residual errors.

The procedure used is as follows:

1. Initial estimates for the parameters are provided.
2. The system equation 5.29 is integrated numerically using the NAG routine *D02EAF*.
3. The function

$$f = NADH_{exp} - NADH_{theo} \tag{5.30}$$

is computed.

	Parameters	Published	Experimental(a)	Experimental(b)
AAT	K_M^{asp}	2.00×10^{-3}	$2.011 \pm 0.170 \times 10^{-3}$	$2.000 \pm 0.000 \times 10^{-3}$
	$K_M^{\alpha\text{-kg}}$	1.00×10^{-4}	$1.285 \pm 0.005 \times 10^{-4}$	$1.275 \pm 0.016 \times 10^{-4}$
	K_M^{oaa}	4.00×10^{-5}	$4.640 \pm 1.250 \times 10^{-5}$	$4.001 \pm 0.001 \times 10^{-5}$
	K_M^{glu}	4.00×10^{-3}	$4.000 \pm 0.000 \times 10^{-3}$	$4.000 \pm 0.000 \times 10^{-3}$
MDH	K_M^{oaa}	4.13×10^{-5}	$8.943 \pm 3.560 \times 10^{-5}$	$4.203 \pm 0.007 \times 10^{-5}$
	K_M^{NADH}	1.70×10^{-5}	$5.631 \pm 3.790 \times 10^{-5}$	$1.700 \pm 0.000 \times 10^{-5}$
	K_M^{mal}	4.15×10^{-3}	$4.149 \pm 0.001 \times 10^{-3}$	$4.150 \pm 0.000 \times 10^{-3}$
	K_M^{NAD}	3.03×10^{-4}	$3.131 \pm 1.460 \times 10^{-4}$	$3.030 \pm 0.000 \times 10^{-4}$
	Sum of squ.		8.856×10^{-11}	1.012×10^{-10}

Table 5.12: Fitting of the Michaelis constants to experimental data for AAT-MDH coupled system. (a) The system is not at steady-state. (b) The system is at steady-state(i.e, $v_1 = v_2$). The NAG routine $E04CCF$ (simplex method) was employed for finding the minimum of the function. Error exit (IFAIL) was 0 when the process was completed. Units for sum of squares are M^2.

4. The optimisation technique (the NAG routine $E04FDF$ or $E04CCF$) is used to find a smaller value of

$$F = \sum f^2. \tag{5.31}$$

5. The process is iterated. It is terminated either after a predeterminated number of iterations, or when F becomes approximately constant over several iterations.

When we tried to fit all nineteen parameters at once minimisation was not successful, because of it took so much computer time and any minimum was lost in experimental noise. It was essential to ignore four V_{max}'s since their published value were used in the first place to assay enzyme activity. Therefore we have 15 parameters, eight of them belong to MDH and other seven belong to AAT.

Even then we have 15 parameters, and it was not feasible to fit all 15 parameters from a single experiment NAG routine failed to converge. However, we assumed some parameters hold their published values, and then we fitted the remaining parameters with some degree of success. The results are displayed in tables 5.12,5.13, 5.14, 5.15, 5.16, 5.17, and 5.18.

	Parameters	Published	Experimental(a)	Experimental(b)
AAT	K_M^{asp}	2.00×10^{-3}	$2.000 \pm 0.000 \times 10^{-3}$	$2.000 \pm 0.000 \times 10^{-3}$
	$K_M^{\alpha\text{-kg}}$	1.00×10^{-4}	$1.270 \pm 0.015 \times 10^{-4}$	$1.270 \pm 0.015 \times 10^{-4}$
	K_M^{oaa}	4.00×10^{-5}	$4.000 \pm 0.000 \times 10^{-5}$	$4.000 \pm 0.000 \times 10^{-5}$
	K_M^{glu}	4.00×10^{-3}	$4.000 \pm 0.000 \times 10^{-3}$	$4.000 \pm 0.000 \times 10^{-3}$
MDH	K_M^{oaa}	4.13×10^{-5}	$4.145 \pm 0.002 \times 10^{-5}$	$4.145 \pm 0.002 \times 10^{-5}$
	K_M^{NADH}	1.70×10^{-5}	$1.700 \pm 0.000 \times 10^{-5}$	$1.700 \pm 0.000 \times 10^{-5}$
	K_M^{mal}	4.15×10^{-3}	$4.149 \pm 0.001 \times 10^{-3}$	$4.149 \pm 0.001 \times 10^{-3}$
	K_M^{NAD}	3.03×10^{-4}	$3.030 \pm 0.000 \times 10^{-4}$	$3.030 \pm 0.000 \times 10^{-4}$
	Sum of squ.		9.947×10^{-10}	9.947×10^{-10}

Table 5.13: Fitting of the Michaelis constants to experimental data for AAT-MDH coupled system. (a) using the NAG routine $E04FCF$ (least squares specialised routine). (b) using the NAG routine $E04FDF$ (least squares specialised routine). Error exit (IFAIL) was 0 when the process was completed. Units for sum of squares are M^2.

	Parameters	Published	Experimental(a)	Experimental(b)
AAT	K_I^{asp}	2.50×10^{-3}	$2.500 \pm 0.000 \times 10^{-3}$	$2.500 \pm 0.000 \times 10^{-3}$
	K_I^{oaa}	3.60×10^{-5}	$3.599 \pm 0.001 \times 10^{-5}$	$3.537 \pm 0.122 \times 10^{-5}$
	K_I^{glu}	6.70×10^{-3}	$6.700 \pm 0.000 \times 10^{-3}$	$6.700 \pm 0.000 \times 10^{-3}$
MDH	K_I^{oaa}	5.00×10^{-4}	$3.843 \pm 0.930 \times 10^{-4}$	$5.000 \pm 0.000 \times 10^{-4}$
	K_I^{NADH}	5.15×10^{-6}	$6.911 \pm 0.340 \times 10^{-6}$	$5.372 \pm 0.029 \times 10^{-6}$
	K_I^{mal}	1.68×10^{-1}	$1.690 \pm 0.006 \times 10^{-1}$	$1.680 \pm 0.000 \times 10^{-1}$
	K_I^{NAD}	1.04×10^{-3}	$6.912 \pm 1.820 \times 10^{-3}$	$1.040 \pm 0.000 \times 10^{-3}$
	Sum of squ.		4.250×10^{-10}	1.304×10^{-9}

Table 5.14: Fitting of the inhibition constants to experimental data for AAT-MDH coupled system; (a) The system is not at steady-state; (b) The system is at steady-state(i.e, $v_1 = v_2 = \mathbf{V}$). The NAG routine $E04CCF$ (simplex method) was employed for finding the minimum of the function. Error exit (IFAIL) was 0 when the process was completed. Units for sum of squares are M^2.

	Parameters	Published	Experimental(a)	Experimental(b)
AAT	K_M^{asp}	2.00×10^{-3}	$2.010 \pm 0.008 \times 10^{-3}$	$2.000 \pm 0.000 \times 10^{-3}$
	$K_M^{\alpha\text{-kg}}$	1.00×10^{-4}	$1.275 \pm 0.038 \times 10^{-4}$	$1.282 \pm 0.070 \times 10^{-4}$
	K_M^{oaa}	4.00×10^{-5}	$5.240 \pm 0.208 \times 10^{-5}$	$4.000 \pm 0.000 \times 10^{-5}$
	K_M^{glu}	4.00×10^{-3}	$4.000 \pm 0.000 \times 10^{-3}$	$4.000 \pm 0.000 \times 10^{-3}$
	K_I^{asp}	2.50×10^{-3}	$1.204 \pm 1.069 \times 10^{-3}$	$2.500 \pm 0.000 \times 10^{-3}$
	K_I^{oaa}	3.60×10^{-5}	$1.400 \pm 0.965 \times 10^{-5}$	$3.598 \pm 0.029 \times 10^{-5}$
	K_I^{glu}	6.70×10^{-3}	$6.701 \pm 0.000 \times 10^{-3}$	$6.700 \pm 0.000 \times 10^{-3}$
	Sum of squ.		7.875×10^{-10}	9.531×10^{-11}

Table 5.15: Comparison of fitting of the kinetic parameters to experimental data for AAT-MDH; (a)There is no relationship between v_1 and v_2; (b) The system is at steady-state($v_1 = v_2 = \mathbf{V}$). The NAG routine $E04CCF$ (simplex method) was employed for finding the minimum of the function. Error exit (IFAIL) was 0 when the process was completed. Units for sum of squares are M^2.

	Parameters	Published value	Experimental value
AAT	K_M^{asp}	2.00×10^{-3}	$2.000 \pm 0.000 \times 10^{-3}$
	$K_M^{\alpha\text{-kg}}$	1.00×10^{-4}	$1.265 \pm 0.023 \times 10^{-4}$
	K_M^{oaa}	4.00×10^{-5}	$4.000 \pm 0.000 \times 10^{-5}$
	K_M^{glu}	4.00×10^{-3}	$4.000 \pm 0.000 \times 10^{-3}$
	K_I^{asp}	2.50×10^{-3}	$2.500 \pm 0.000 \times 10^{-3}$
	K_I^{oaa}	3.60×10^{-5}	$3.592 \pm 0.018 \times 10^{-5}$
	K_I^{glu}	6.70×10^{-3}	$6.700 \pm 0.000 \times 10^{-3}$
	Sum of squ.	9.512×10^{-11}	9.512×10^{-11}

Table 5.16: Fitting of the kinetic parameters to experimental data for AAT-MDH; The NAG least squares specialised routine $E04FDF$ was used. Error exit (IFAIL) was 0 when the process was completed. Units for sum of squares are M^2.

	Parameters	Published	Experimental(a)	Experimental(b)
MDH	K_M^{oaa}	4.13×10^{-5}	$7.042 \pm 1.642 \times 10^{-5}$	$5.694 \pm 1.550 \times 10^{-5}$
	K_M^{NADH}	1.70×10^{-5}	$3.008 \pm 1.304 \times 10^{-5}$	$1.705 \pm 0.137 \times 10^{-5}$
	K_M^{mal}	4.15×10^{-3}	$2.243 \pm 0.338 \times 10^{-3}$	$4.150 \pm 0.000 \times 10^{-3}$
	K_M^{NAD}	3.03×10^{-4}	$3.125 \pm 0.053 \times 10^{-4}$	$3.030 \pm 0.000 \times 10^{-4}$
	K_I^{oaa}	5.00×10^{-4}	$3.600 \pm 0.700 \times 10^{-4}$	$5.000 \pm 0.000 \times 10^{-4}$
	K_I^{NADH}	5.15×10^{-6}	$6.740 \pm 0.540 \times 10^{-6}$	$7.475 \pm 2.860 \times 10^{-6}$
	K_I^{mal}	1.68×10^{-1}	$1.689 \pm 0.049 \times 10^{-1}$	$1.680 \pm 0.000 \times 10^{-1}$
	K_I^{NAD}	1.04×10^{-3}	$7.374 \pm 2.334 \times 10^{-3}$	$1.040 \pm 0.000 \times 10^{-4}$
	Sum of squ.		5.106×10^{-9}	1.141×10^{-10}

Table 5.17: Comparison of fitting of the kinetic parameters to experimental data for AAT-MDH; (a)There is no relationship between v_1 and v_2; (b) The system is at steady-state($v_1 = v_2 = \mathbf{V}$). The NAG routine $E04CCF$ (simplex) was employed for finding the minimum of the function. Error exit (IFAIL) was 0 when the process was completed. Units for sum of squares are M^2.

	Parameters	Published value	Experimental value
MDH	K_M^{oaa}	4.13×10^{-5}	$5.690 \pm 0.987 \times 10^{-5}$
	K_M^{NADH}	1.70×10^{-5}	$1.700 \pm 0.007 \times 10^{-5}$
	K_M^{mal}	4.15×10^{-3}	$4.150 \pm 0.000 \times 10^{-3}$
	K_M^{NAD}	3.03×10^{-4}	$3.030 \pm 0.000 \times 10^{-4}$
	K_I^{oaa}	5.00×10^{-4}	$5.000 \pm 0.000 \times 10^{-4}$
	K_I^{NADH}	5.15×10^{-6}	$6.769 \pm 1.967 \times 10^{-6}$
	K_I^{mal}	1.68×10^{-1}	$1.680 \pm 0.000 \times 10^{-1}$
	K_I^{NAD}	1.04×10^{-3}	$1.040 \pm 0.000 \times 10^{-4}$
	Sum of squ.		9.987×10^{-11}

Table 5.18: Fitting of the kinetic parameters to experimental data for aspartate aminotransferase-malate dehydrogenase coupled system. The NAG least squares specialised routine $E04FDF$ was employed for finding the minimum of the sum of squares. Error exit (IFAIL) was 0 when the process was completed. Units for sum of squares are M^2.

5.2.5 Improvements to the Direct Fitting of Kinetic Parameters

We have already pointed out that it is impossible to fit all kinetic constants from single time-course data [Bayram, Bennett and Dewar, 1991]. Published values of V_{max} are used in the assaying of the enzyme concentration, and so cannot then themselves be determined by experiments depending on that enzyme concentration. We have previously shown that reliable estimates of the 8 Michaelis constants for our experimental system could be obtained from the data, where published values were used for all other kinetic parameters.

However a better approach is to look at the sensitivity of the overall rate law to variations in the kinetic parameters. For each parameter, k_i, we wish to know how a small relative change in k_i will influence V. In other words

$$lim \frac{\delta V/V}{\delta k_i/k_i} = \frac{\partial lnV}{\partial lnk_i}$$

i.e. we determine the scaled derivative such that:

$$C^V_{k_i} = \frac{k_i}{V} \times \frac{\partial V}{\partial k_i}$$

taking all other parameters at their published value. We do not know the experimental error in V directly, but this can be estimated from the known errors in $[NADH]$ and t because values for the scaled derivatives in fact vary through the time course of the experiment, since they are a function of $[NADH]$ concentration. We can then decide which kinetic parameters can be estimated with low error by this technique. A large value of the scaled derivative indicates that fitting k_i by minimising residuals in V will lead to good estimates of k_i. Any small error in k_i will greatly affect V (see figure 5-10). Correspondingly where the scaled derivative is small, variation in the estimate for k_i will have little effect on V (see figure 5-9), and so the estimate is unlikely to be reliable.

To calculate scaled derivative of the each kinetic parameters in the system aspartate aminotransferase-malate dehydrogenase, we left the kinetic parameters in symbolic form

in the rate laws, v_1 and v_2. Five conservation equations, 4.34 were solved using following REDUCE format;

```
    sol := solve({ eq1, eq2, eq3, eq4, eq5 },
                 { asp, akg, glu, nad, mal } )$
        asp := rhs ( part ( sol, 1 ) ) $
        akg := rhs ( part ( sol, 2 ) ) $
        glu := rhs ( part ( sol, 3 ) ) $
        nad := rhs ( part ( sol, 4 ) ) $
        mal := rhs ( part ( sol, 5 ) ) $
% Now substitute these value into the rate laws.
        equ6 := num( V - v1 ) $
        equ7 := num( V - v2 ) $
% equ7 is linear in oaa.
        der := deg( equ7, oaa ) $
        if der = 1 then
           oaa := part( solve ( equ7, oaa ), 1, 2 ) $
% substitute oaa in the equ6 and obtain flux equation in terms of
% NADH and kinetic parameters.
        fun := num( equ6 ) $
```

The equation **fun** was cubic in **V** so that we calculated the implicit derivative of **V** with respect to each kinetic parameters using following REDUCE format;

```
    for i := 1 : 15 do
        << depend V, k(i) >> $
    for i := 1 : 15 do
        << deriv(i) := df( fun, k(i) ) $
           dvdk(i)  := -coeffn( deriv(i), df(V, k(i)), 0 )/
                       coeffn( deriv(i), df(V, k(i)), 1 )>> $
```

The equation **fun** was solved using NAG routine $C02AEF$ and meaningful root was obtained (too large to do in REDUCE). By substituting the meaningful root into

dvdk(i) we found numerical value of **dvdk(i)**. We repeated this process for each kinetic parameter. Thus scaled derivatives of each of the parameters were calculated. At the starting concentration we obtained the scaled derivatives of the kinetic parameters of the aspartate aminotransferase-malate dehydrogenase system, which are given in table 5.19. The parameters were shown in table 5.19 by (*) are very well determined; others are very poorly determined, in many cases the actual errors are very high for these parameters. The scaled derivative accurately tells which parameters should have large errors. One of the most significant thing to learn from the scaled derivative of the parameters are that even for systems with many data point (scaled derivatives are calculated at just one data points), some parameters may be very poorly determined. According to the scaled derivatives of the kinetic parameters we grouped the parameters to fit, and the results are given in tables 5.20 and 5.22. As can be seen from the table 5.20 overall flux equation, **V**, much more depends on the kinetic parameters, $K_{\mathrm{M}}^{\alpha\text{-kg}}$ and $K_{\mathrm{I}}^{\mathrm{oaa}}$ in the system AAT, and $K_{\mathrm{M}}^{\mathrm{oaa}}$ and $K_{\mathrm{I}}^{\mathrm{NADH}}$ in the system MDH, rather than others. However, the values of the kinetic parameters obtained (see table 5.20) have a low standard deviation error by bootstrap suggesting they fitted the data well. The result shows that the use of the scaled derivative method to estimate the kinetic parameters significantly improved the accuracy and the expected accuracy in determining the parameters in the given system. A statistical distribution of the parameters is shown in the figures 5-2, 5-3, 5-4 and 5-5.

5.2.6 Interrelationships of Kinetic Parameters

When we have a set of bivariate data in which k_j and k_k appear to be related and the relationship appears to be rectilinear, as may wish to assess the closeness of the apparent relationship and test its significance. This may be done by using the method of correlation. We need a measure of the degree to which k_j and k_k vary together. One such measure is the sum of the products of the joint deviations of k_j and k_k from their respective means, divided by the number of degrees of freedom, i.e. the covariance (C_{jk}) between k_j and k_k, thus:

$$C_{jk} = \frac{\sum_{i=1}^{n}(k_{ij} - \bar{k}_j) \times (k_{ik} - \bar{k}_k)}{n-1} \tag{5.32}$$

	Parameter	Scaled derivative	Parameter	Scaled derivative
AAT	K_M^{asp}	-1.768×10^{-05}	K_I^{asp}	-3.170×10^{-04}
	$*K_M^{\alpha\text{-kg}}$	-9.913×10^{-01}	$*K_I^{oaa}$	4.029×10^{-01}
	K_M^{oaa}	-2.765×10^{-04}	K_I^{glu}	2.594×10^{-07}
	K_M^{glu}	3.153×10^{-04}		
MDH	$*K_M^{oaa}$	-1.368×10^{-01}	K_I^{oaa}	-5.879×10^{-06}
	K_M^{NADH}	-1.546×10^{-07}	$*K_I^{NADH}$	-7.877×10^{-01}
	K_M^{mal}	7.267×10^{-02}	K_I^{mal}	2.475×10^{-10}
	K_M^{NAD}	-4.455×10^{-05}	K_I^{NAD}	7.324×10^{-03}

Table 5.19: Scaled derivatives for aspartate aminotransferase malate dehydrogenase coupled system.

The covariance is positive when k_j and k_k tend to vary together and negative when one tends to increase as the other decreases, but it is not a suitable measure of the closeness of the relationship because its magnitude depends on the units in which k_j and k_k are measured. This disadvantage is removed by expressing the deviation in standard deviation units. The resulting quantity is called the *product-moment correlation coefficient* [Taylor, 1982] and our estimate of it is designated r_{jk}, thus:

$$\begin{aligned} r_{jk} &= \frac{\sum_{i=1}^{n}(k_{ij}-\bar{k}_j)(k_{ik}-\bar{k}_k)}{\sqrt{(\sum_{i=1}^{n}(k_{ij}-\bar{k}_j)^2 \sum_{i=1}^{n}(k_{ik}-\bar{k}_k)^2)}} \\ j,k &= 1,\cdots,5 \ (number\ of\ parameters) \\ and\ n &= 100 \ (number\ of\ cases) \end{aligned} \tag{5.33}$$

and the correlation coefficient r_{jk} can have values ranging from -1 to $+1$. $r_{jk} = +1$ corresponds to a rectilinear relationship in which the two variables are positively related. The case $r_{jk} = -1$ corresponds to a rectilinear relationship in which the two variables are negatively related. A value of r_{jk} near -1 and $+1$ indicate an approach to a rectilinear relationship. If value of r_{jk} is close to 0, then the two variables are uncorrelated. The correlation coefficient between the kinetic parameters for AAT-MDH coupled system were given in table 5.21.

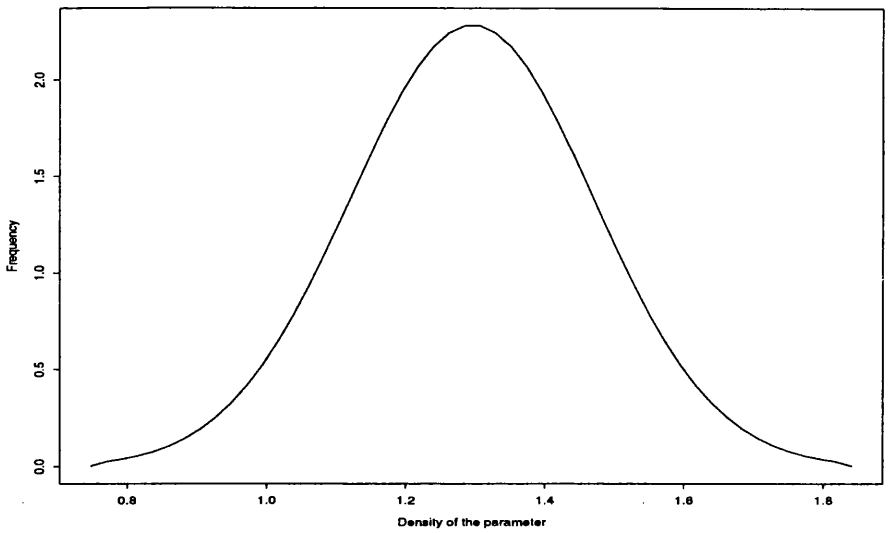


Figure 5-2: A distribution of the kinetic parameter $K_M^{\alpha\text{-kg}}$ in the aspartate aminotransferase-malate dehydrogenase coupled system.

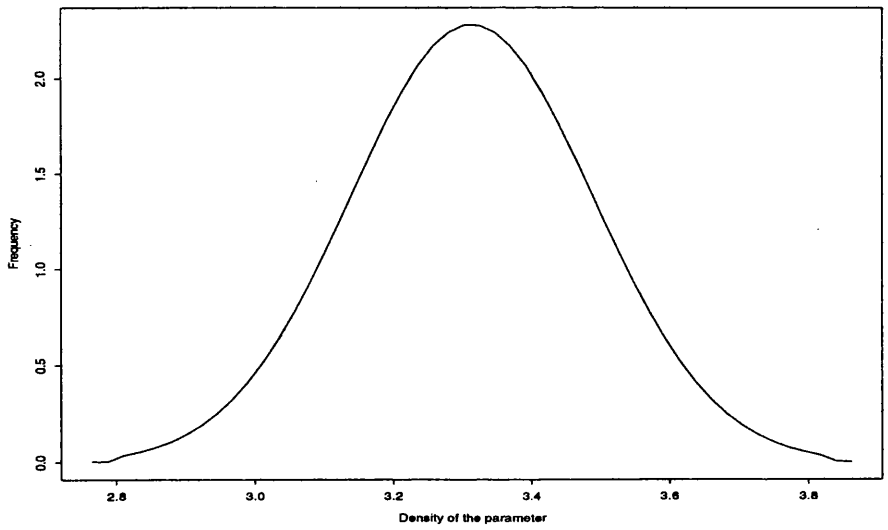


Figure 5-3: A distribution of the kinetic parameter K_I^{oaa} in the aspartate aminotransferase-malate dehydrogenase coupled system.

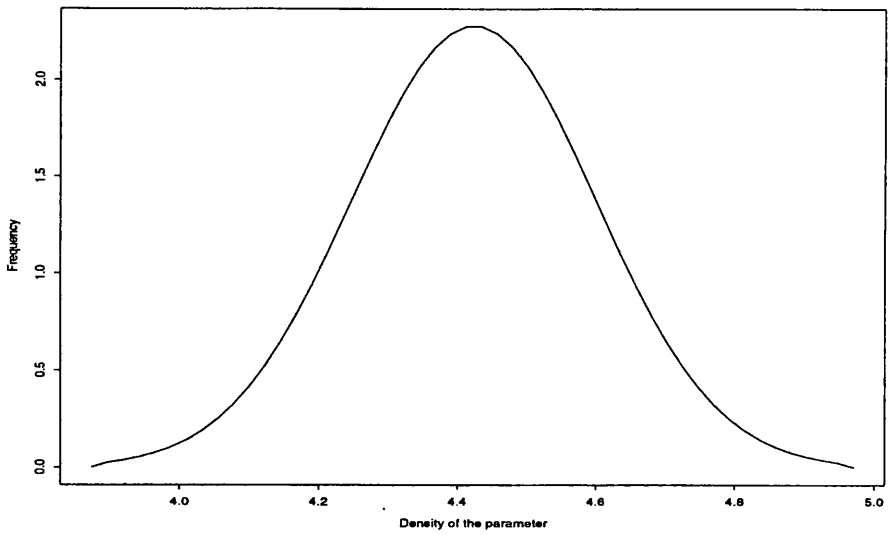


Figure 5-4: A distribution of the kinetic parameter K_M^{oaa} in the aspartate aminotransferase-malate dehydrogenase coupled system.

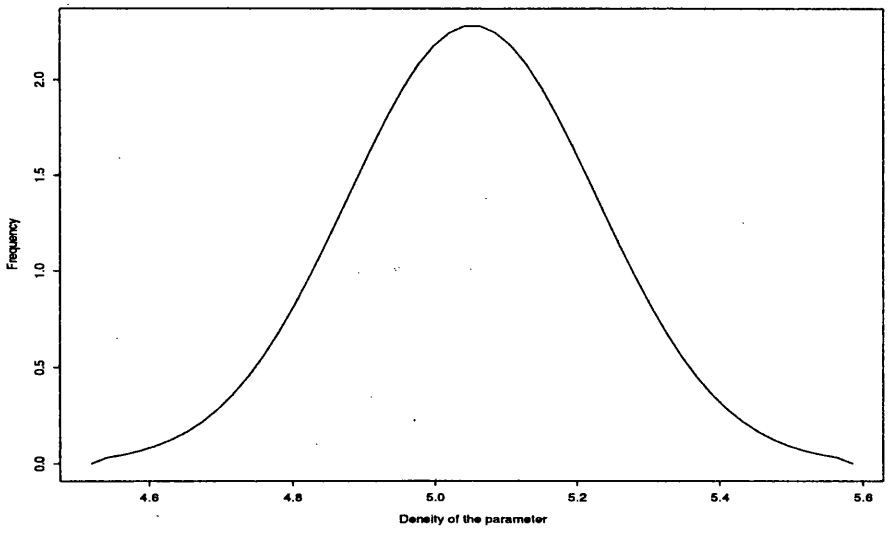


Figure 5-5: A distribution of the kinetic parameter K_I^{NADH} in the aspartate aminotransferase-malate dehydrogenase coupled system.

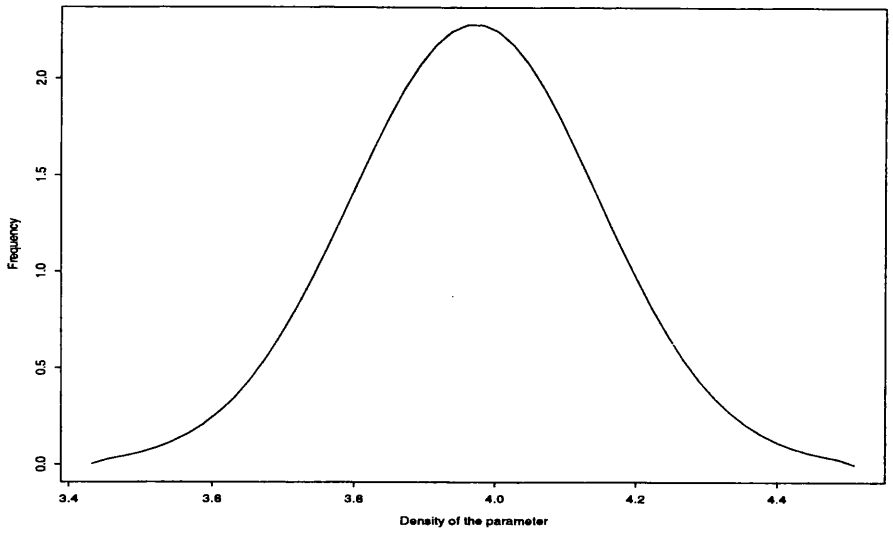


Figure 5-6: A distribution of the kinetic parameter K_M^{mal} in the aspartate aminotransferase-malate dehydrogenase coupled system.

	Parameters	Published	Experimental (a)	Experimental (b)
AAT	$K_{\mathrm{M}}^{\alpha\text{-kg}}$	1.00×10^{-4}	$1.2949 \pm 0.0212 \times 10^{-4}$	$1.1120 \pm 0.0129 \times 10^{-4}$
	$K_{\mathrm{I}}^{\mathrm{oaa}}$	3.60×10^{-5}	$3.3140 \pm 0.0234 \times 10^{-5}$	$3.3151 \pm 0.0214 \times 10^{-5}$
MDH	$K_{\mathrm{M}}^{\mathrm{oaa}}$	4.13×10^{-5}	$4.4235 \pm 0.0221 \times 10^{-5}$	$4.4320 \pm 0.0014 \times 10^{-5}$
	$K_{\mathrm{I}}^{\mathrm{NADH}}$	5.15×10^{-6}	$5.0526 \pm 0.0074 \times 10^{-6}$	$4.7410 \pm 0.0244 \times 10^{-6}$
	$K_{\mathrm{M}}^{\mathrm{mal}}$	4.15×10^{-3}	$3.9706 \pm 0.0136 \times 10^{-3}$	$3.9505 \pm 0.0123 \times 10^{-3}$
	Sum of squares.		1.687×10^{-10}	1.657×10^{-10}

Table 5.20: Fitting of the kinetic parameters to experimental data for an aspartate aminotransferase-malate dehydrogenase coupled system. Errors estimate has been done by bootstrap. Experimental value = $\bar{k}_i \pm$ *Standard error.* (a) using the NAG routine $E04CCF$ (simplex method), (b) using the NAG least squares specialised routine $E04FDF$. Error exit (IFAIL) was 0 when the process was completed. Units for sum of squares are M^2.

parameters	$K_M^{\alpha\text{-kg}}$	K_I^{oaa}	K_M^{oaa}	K_I^{NADH}	K_M^{mal}
$K_M^{\alpha\text{-kg}}$	1.0000	−0.8533	0.9790	−0.9772	−0.9963
K_I^{oaa}	−0.8533	1.0000	−0.7290	0.9428	0.8939
K_M^{oaa}	0.9790	−0.7290	1.0000	−0.9139	−0.9583
K_I^{NADH}	−0.9772	0.9428	−0.9139	1.0000	0.9912
K_M^{mal}	−0.9963	0.8939	−0.9583	0.9912	1.0000

Table 5.21: Correlation coefficients between kinetic parameters.As can be seen all correlation coefficients are very close to −1 and +1 that indicate the kinetic parameters are correlated.

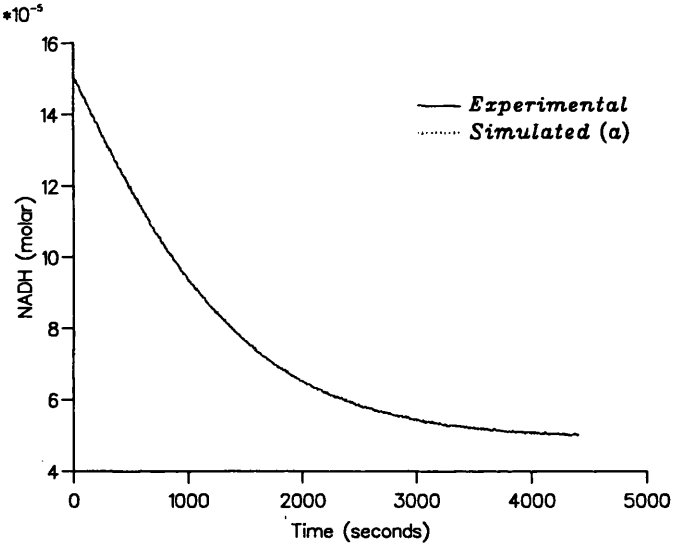


Figure 5-7: Graphical form of the output from the simulation; (a) obtained after fitting the kinetic parameters (table 5.20, experimental (a)) to experimental data.

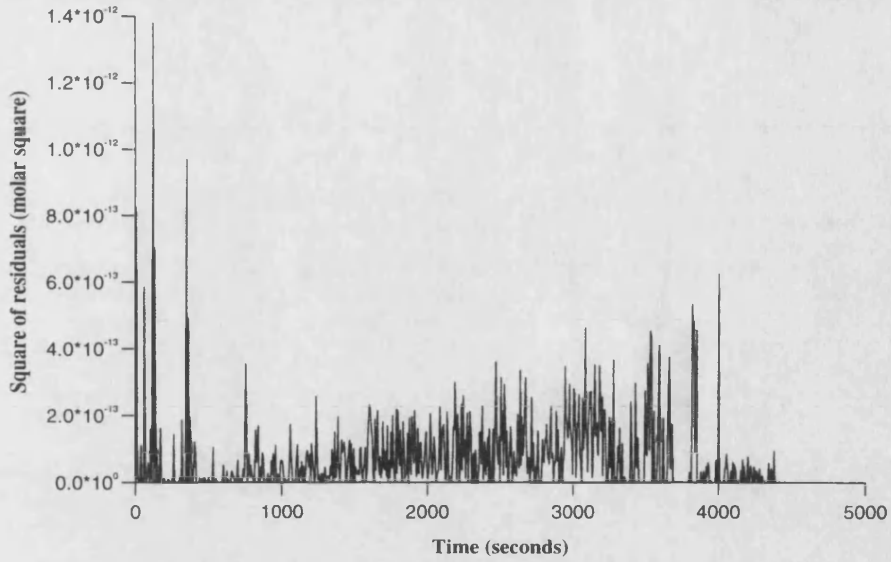


Figure 5-8: Graphical form of squares of residuals against time. The residuals were recorded as differences between the numerical and experimental value of NADH. The numerical values of NADH were calculated by integrating steady state rate law, V, in which fitted kinetic parameters (table 5.20) were used. The graph shows that our parameter fitting are generally good, and accuracy is comparable to the expected accuracy.

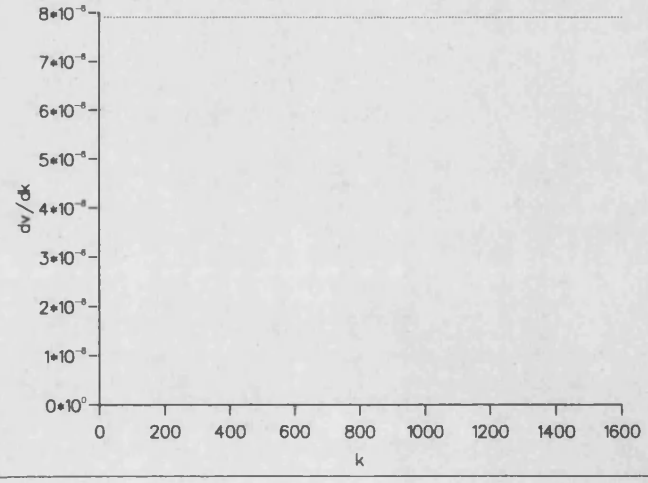


Figure 5-9: Effect of changes in the kinetic parameter k_2 on the flux function v in the MDH-FUM. As can be seen, it is not important whether the value of k_2 is small or big, because it has no influence v.

	Parameters	Published	Experimental (a)	Experimental (b)
MDH	K_M^{NADH}	1.70×10^{-5}	$1.687 \pm 0.116 \times 10^{-5}$	$1.578 \pm 0.234 \times 10^{-5}$
	K_M^{NAD}	3.03×10^{-4}	$3.063 \pm 0.294 \times 10^{-4}$	$3.056 \pm 0.301 \times 10^{-4}$
	K_I^{oaa}	5.00×10^{-4}	$4.987 \pm 0.116 \times 10^{-4}$	$4.980 \pm 0.109 \times 10^{-4}$
	K_I^{mal}	1.68×10^{-1}	$1.713 \pm 0.295 \times 10^{-1}$	$1.606 \pm 0.198 \times 10^{-1}$
	K_I^{NAD}	1.04×10^{-3}	$1.027 \pm 0.162 \times 10^{-3}$	$1.023 \pm 0.153 \times 10^{-3}$
AAT	K_M^{asp}	2.00×10^{-3}	$2.033 \pm 0.294 \times 10^{-3}$	$2.021 \pm 0.302 \times 10^{-3}$
	K_M^{oaa}	4.00×10^{-5}	$3.987 \pm 0.161 \times 10^{-5}$	$3.894 \pm 0.197 \times 10^{-5}$
	K_M^{glu}	4.00×10^{-3}	$4.033 \pm 0.294 \times 10^{-3}$	$4.035 \pm 0.278 \times 10^{-3}$
	K_I^{asp}	2.50×10^{-3}	$2.487 \pm 0.116 \times 10^{-3}$	$2.451 \pm 0.102 \times 10^{-3}$
	K_I^{glu}	6.70×10^{-3}	$6.733 \pm 0.291 \times 10^{-3}$	$6.703 \pm 0.268 \times 10^{-3}$
	Sum of squ.		1.691×10^{-10}	1.569×10^{-10}

Table 5.22: Fitting of the kinetic parameters to experimental data for an aspartate aminotransferase-malate dehydrogenase coupled system. The value of scaled derivatives of above parameters were small. This result indicates that the use of the scaled derivative of the parameters can significantly improve our ability to determine some of the parameters in the system. (a) using the NAG routine $E04CCF$ (simplex method), (b) using the NAG least squares specialised routine $E04FDF$. Error exit (IFAIL) was 0 when the process was completed. Units for sum of squares are M^2.

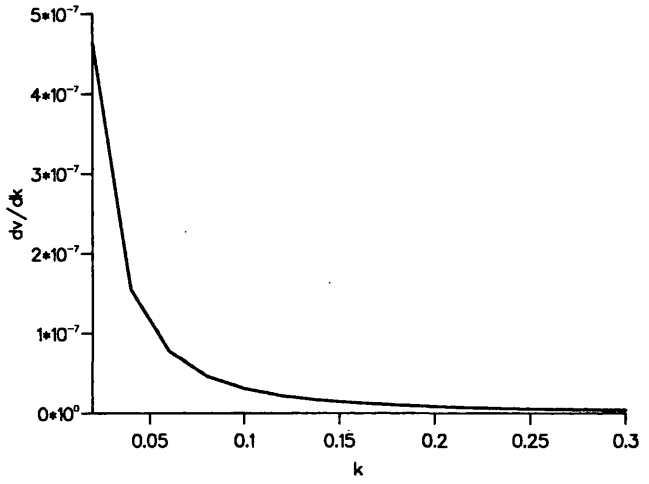


Figure 5-10: Graphical form of k which affects v. Any small changes in k is greatly affect v. It meant that the flux function, v, much more depends on the kinetic parameter k, and the kinetic parameter is fitted to the data generally good.

	Parameter	Calculated val.	Parameter	Calculated val.
MDH	k_1	8.024×10^7	k_5	3.480×10^3
	k_2	4.132×10^2	k_6	4.451×10^5
	k_3	3.749×10^7	k_7	2.243×10^3
	k_4	4.705×10^2	k_8	2.157×10^6

Table 5.23: The numerical values of the individual rate constants for malate dehydrogenase system. The values were calculated from numerical value of V_{max}, K_M and K_I using relationship between k_i and V_{max} ,K_M and K_I.

	parameter	Calculated val.	parameter	Calculated val.
FUM	k_1	6.80×10^7	k_3	9.00×10^2
	k_2	8.00×10^2	k_4	3.40×10^8

Table 5.24: The numerical values of the individual rate constants for fumarase system. The values were calculated from numerical value of the V_{max} and K_M.

5.2.7 Using Individual Rate Constants

We have used the rate laws in terms of Michaelis constants. However we can work directly with the enzyme mechanism and this offers some advantages. For example the advantage in working directly with k_i's in that there are fewer of them. The rate laws for aspartate aminotransferase and malate dehydrogenase have two V_{max}, four K_M and four K_I each; a total of 20 parameters, By implication, working with Michaelis parameters, inhibition constants etc. involves redundant parameters. Indeed if we try to compute a Gröbner Basis to transform values of V_{max}, K_M and K_I into k_i we fail for the system aspartate aminotransferase because the published values are not self-consistent.

In case we can always obtain formulae for V_{max} ,K_M etc, from their functional specification in terms of k_i.

Let us consider a malate dehydrogenase system. The rate law of the system was given in the previous section in terms of k_i and also in terms of V_{max} ,K_M etc. If their published relationships are used than the results are as in table 5.23.

We transformed values of V_{max} and K_M into k_i using computer algebra techniques for a fumarase system and their numerical values are as in table 5.24.

Now we get 12 rate constants for a malate dehydrogenase-fumarase coupled system instead of 14 Michaelis parameters. The overall flux equation of the system is in the following form,

$$v = -\frac{d[NADH]}{dt} = f(NADH, k_i; i = 1, ..., 12) \tag{5.34}$$

We have experimental data for this system, in the form of time-course data for the reaction. This provides us with 217 data points of [NADH] concentration against time as the reaction proceeds. We wish to fit our overall rate law to this, to obtain estimates of the kinetic parameters k_i. At this stage we resort to numerical techniques and use the established technique to obtain estimates [Bock 1981; Bayram, Bennett and Dewar, 1991].

At the end of data point the values of the scaled derivative of the parameters were calculated using similar method described in the previous section, and displayed in table 5.25. The values for the scaled derivatives in fact vary through the time course of the experiment, since they are a function of the [NADH] concentration. From table 5.25 we selected five parameters to fit the overall rate law using experimental data. The result is given in table 5.26. The simulation using fitted rate constants is displayed in figures 5-11 and 5-12. The figure 5-12 shows that there are not much more agreement between experimental and simulated $NADH$ time course data. They should be fitted because we used rate constants which were fitted to experimental data. However, when we work directly with the enzyme mechanism, we can not select some rate constant using the scaled derivative method because all rate constants depend on each other [D. Fell; personal communication], i.e. a small relative change in one of the rate constants influences the others. We have fitted all rate constants at one time for a malate dehydrogenase isolated system, and performed simulation using them. The simulation showed that the parameters fit to the data generally very well. The results were shown in the table 5.28, figure 5-13 and 5-14.

	Parameter	Scaled deriv.	Parameter	Scaled deriv.
MDH	$*k_1$	1.182×10^{-00}	$*k_5$	1.928×10^{-00}
	k_2	2.053×10^{-03}	k_6	9.015×10^{-04}
	$*k_3$	1.996×10^{-00}	$*k_7$	1.870×10^{-00}
	k_4	4.269×10^{-04}	k_8	1.928×10^{-02}
FUM	k_1	-4.996×10^{-01}	k_3	1.714×10^{-04}
	k_2	3.584×10^{-05}	$*k_4$	2.499×10^{-00}

Table 5.25: Scaled derivatives for a malate dehydrogenase-fumarase coupled system at the end of the experimental data point. The system is at the steady state. We belie that the parameters(*) have influence the overall flux (v).

	Parameter	Calculated val.	Experimental (a)	Experimental (b)
MDH	k_1	8.024×10^7	$3.782 \pm 0.052 \times 10^7$	$3.782 \pm 0.052 \times 10^7$
	k_3	3.749×10^7	$6.694 \pm 0.132 \times 10^7$	$6.691 \pm 0.125 \times 10^7$
	k_5	3.480×10^3	$5.811 \pm 0.763 \times 10^3$	$5.801 \pm 0.221 \times 10^3$
	k_7	2.243×10^3	$6.503 \pm 0.457 \times 10^3$	$6.497 \pm 0.369 \times 10^3$
FUM	k_4	3.400×10^8	$7.795 \pm 2.122 \times 10^8$	$7.651 \pm 2.084 \times 10^8$
	Sum of squares		9.849×10^{-13}	9.811×10^{-13}

Table 5.26: Fitting individual rate constants to experimental data for malate dehydrogenase-fumarase coupled system. The rate constants were fitted using bootstrap method. (a) using the NAG routine $E04CCF$ (simplex method), (b) using the NAG least squares specialised routine $E04FDF$. Error exit (IFAIL) was 0 when the process was completed. Units for sum of squares are M^2.

	Parameter	Scaled deriv.	Parameter	Scaled deriv.
MDH	k_1	8.044×10^{-01}	k_5	7.391×10^{-01}
	k_2	-8.699×10^{-02}	k_6	-2.467×10^{-03}
	k_3	1.426×10^{-01}	k_7	1.552×10^{-01}
	k_4	-1.750×10^{-02}	k_8	-6.924×10^{-02}

Table 5.27: Scaled derivatives for a malate dehydrogenase isolated system. As can be seen from the table almost all rate constants have influence on the rate law apart from k_6.

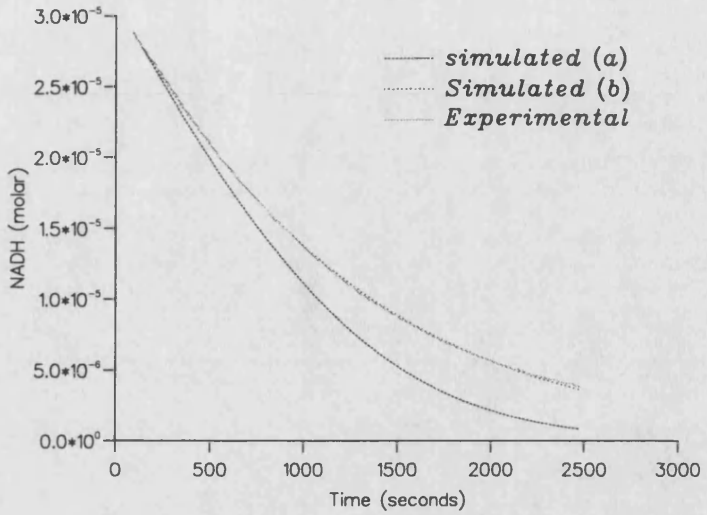


Figure 5-11: Graphical form of the output from the simulation for MDH-FUM; (a) calculated using published rate constants; (b) calculated using fitted rate constants.

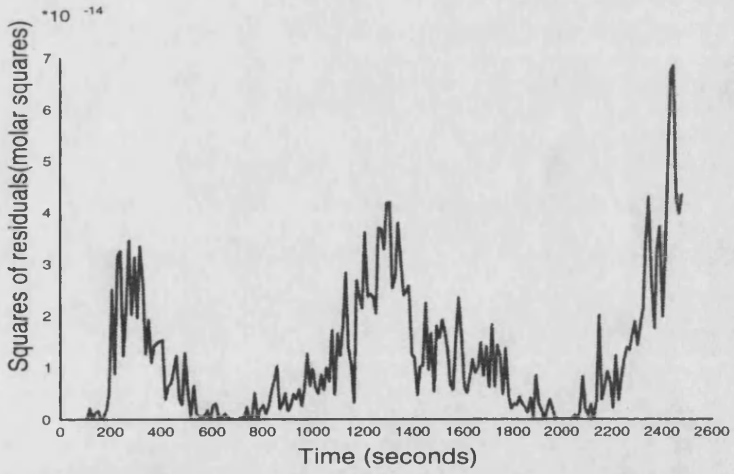


Figure 5-12: Graphical form of the squares of the residuals against to the time for malate dehydrogenase-fumarase coupled system.

	Parameters	Calculated value	Experimental (a)	Experimental(b)
MDH	k_1	8.024×10^7	$5.390 \pm 0.587 \times 10^8$	$5.387 \pm 0.581 \times 10^8$
	k_2	4.132×10^2	$3.995 \pm 0.867 \times 10^4$	$3.996 \pm 0.863 \times 10^4$
	k_3	3.749×10^7	$6.761 \pm 0.570 \times 10^8$	$6.762 \pm 0.567 \times 10^8$
	k_4	4.705×10^2	$2.239 \pm 2.365 \times 10^3$	$2.237 \pm 2.360 \times 10^3$
	k_5	3.480×10^3	$4.668 \pm 0.890 \times 10^3$	$4.661 \pm 0.886 \times 10^3$
	k_6	4.451×10^5	$9.280 \pm 4.829 \times 10^6$	$9.281 \pm 4.827 \times 10^6$
	k_7	2.243×10^3	$1.326 \pm 0.311 \times 10^3$	$1.321 \pm 0.300 \times 10^3$
	k_8	2.157×10^6	$9.859 \pm 0.570 \times 10^7$	$9.842 \pm 0.531 \times 10^7$
	Sum of squ.		6.926×10^{-11}	6.871×10^{-11}

Table 5.28: Fitting of the rate constants to experimental data for malate dehydrogenase isolated system. The rate constants were fitted using bootstrap method. (a) using the NAG routine $E04CCF$ (simplex method), (b) using the NAG least squares specialised routine $E04FDF$. Error exit (IFAIL) was 0 when the process was completed. Units for sum of squares are M^2.

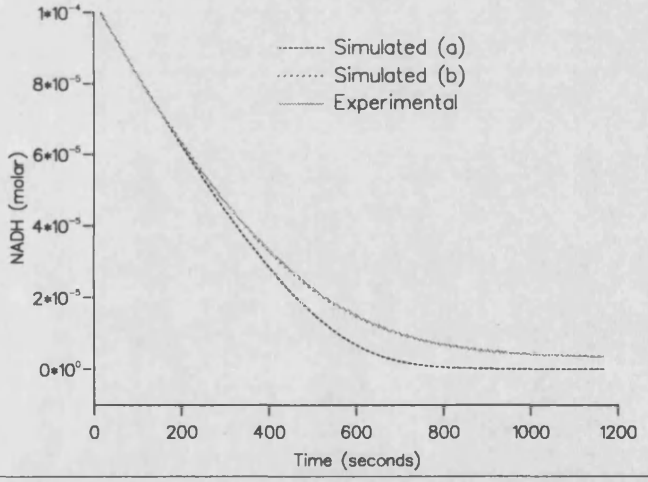


Figure 5-13: Graphical form of the output from the simulation for malate dehydrogenase isolated system; (a) using published value of rate constants; (b) using fitted value of rate constants.

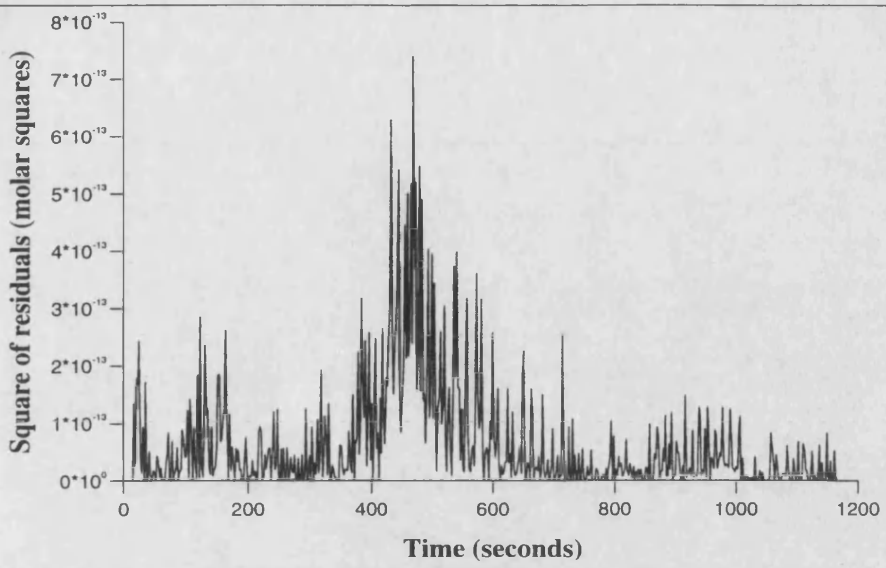


Figure 5-14: Graphical form of the squares of residuals against to the time; This graph shows that our parameters fit generally very well.

	Parameter	calculated val.	Experimental val.
MDH	k_2	4.132	$4.132 \pm (5.098 \times 10^{-7})$
	k_4	4.705	$4.705 \pm (7.209 \times 10^{-7})$
	k_6	4.451	$4.451 \pm (6.743 \times 10^{-7})$
	k_8	2.157	$2.157 \pm (3.372 \times 10^{-7})$
FUM	k_1	6.800	$6.800 \pm (1.380 \times 10^{-8})$
	k_2	8.000	$1.797 \pm (5.517 \times 10^{-0})$
	k_3	9.000	$9.000 \pm (1.490 \times 10^{-8})$
	Sum of squ.		5.0321×10^{-11}

Table 5.29: Fitting individual rate constants to experimental data for malate dehydrogenase-fumarase coupled system. As is known, the value of scaled derivatives of the above rate constants were small, so that the estimate is unlikely to be reliable. Before fitting the rate constants to the data, they were scaled such as each of the rate constants had order unity. The NAG routine $E04CCF$ (simplex method) was used for finding the minimum of the function. Error exit (IFAIL) was 0 when the process was completed.

5.2.8 Summary

We have shown in this chapter that computer algebra and numerical analysis (NAG) techniques can solve problems that occur in enzyme kinetics. We have seen how when presented with values of enzyme kinetic parameters. We can obtain general flux equation. This general technique for eliminating unknowns from a system of non-linear simultaneous equations has been turned round, to give us a way of determining enzyme kinetic parameters.

We have also demonstrated that our scaled derivative method for parameter identification has worked well on a coupled enzyme system. In most cases it is able to produce parameter values with low errors.

Chapter 6

Metabolic Control Analysis

The theoretical principles developed by Kacser and Burns [Kacser and Burns, 1973] and Heinrich and Rapoport [Heinrich and Rapoport, 1974] have greatly facilitated the quantitative determination of the extent to which certain enzymes control the flux through a metabolic pathway. Metabolic Control analysis is concerned with the effect of changes in enzyme concentration or activity on the steady state metabolite concentration and fluxes of the metabolic systems. The basic relationships are the summation and connectivity theorems. They allow one to express the behaviour of the system variables in terms of the kinetic properties of the isolated enzymatic reactions that build up the metabolic network. Much work has been done on this subject [Kacser et al, 1990; Acerenza *et al*, 1989; Cascante *et al*, 1990; Small and Fell, 1990; Westerhoff and Chen, 1984]. A matrix method has been derived [Fell and Sauro, 1985; Sauro *et al*, 1987] that allows the determination of the flux and concentration control coefficients of enzymes from their kinetic properties represented by the elasticity coefficients. A survey of metabolic control analysis has been given by D. Fell [Fell, 1992].

Flux and metabolite control coefficients and enzyme elasticities are more useful for many purposes. From the definition of metabolite and flux control coefficients and elasticities we are able to derive symbolic forms of these functions, in terms of conventional kinetic parameters and steady state metabolite concentrations. At the simplest level we are able to substitute values of these kinetic parameters, to yield values for the metabolic control coefficients. Since we are substituting into symbolic

equations we can always guarantee conservation relationships hold [Bayram, Bennett and Dewar, 1991]. This is a major advance on existing measurement methodologies, which rely on genetically manipulating the enzyme concentrations to determine these parameters, with considerable experimental error.

6.1 Flux Control Coefficients

We shall consider the aspartate aminotransferase-malate dehydrogenase coupled system consisting of metabolite *oaa* and two enzymes AAT and MDH.

Let us consider a small change in the activity of any one enzyme (AAT or MDH) within the system. This could be produced by a change in the concentration of the enzyme or in any kinetic parameter of the enzyme. All such changes can always be replaced by a change in the concentration of the enzyme. The result of such a change is to bring about a change in the local reaction rate and, consequently, different small change in the concentrations of the intermediate metabolites associated with each catalysed step because of the coupling between them. The response(δV) of the system flux(V) to a small change (δAAT or δMDH) in any one enzyme in the system will be a consequence of all the changes in metabolite concentrations as well as the change in enzyme activity. This response is defined by the flux control coefficient, i.e, the flux control coefficient is defined with respect to one specified enzyme and for one defined flux in the system.

The flux control coefficient is a measure of flux control, i.e, how an enzyme's activity has influence over the pathway flux.

In a linear pathway, the flux control coefficient can take on any value between zero and one, larger and negative values are possible in non-linear pathways [D. Fell, personal communication]. A value of zero indicates no control and a value of one complete proportional control. For more complex pathways, e.g. branches, or cycles, the flux control coefficient is not limited by this range. It can be greater than one; or less than zero; coefficients less than zero simply indicate that a rise in enzyme activity induces a fall in flux.

To calculate flux control coefficients for the aspartate aminotransferase-malate

dehydrogenase coupled system, we left enzyme concentrations in symbolic form in the rate laws v_1 and v_2. Then a Gröbner basis was calculated using following REDUCE format

```
bas := groebner({ equ1, equ2, equ3, equ4, equ5, equ6, equ7},
                { asp , akg , glu , oaa , mal , nad , v }) $
F := num( part( bas, 7 )) $
```

and we obtained an equation in terms of $\mathbf{V}$, $NADH$ and enzyme concentrations AAT and MDH. The equation was cubic with respect to $\mathbf{V}$, i.e. of the form

$$F = a \times \mathbf{V}^3 + b \times \mathbf{V}^2 + c \times \mathbf{V} + d \tag{6.1}$$

where a, b, c and d are terms in the $NADH$, AAT and MDH concentrations. The first implicit derivatives of F with respect to each enzyme, AAT and MDH, were calculated in REDUCE using following format;

```
depend  v, aat $
depend  v, mdh $
der1 := df( F, aat ) $
der2 := df( F, mdh ) $
```

These two equation were linear with respect to $\partial V/\partial AAT$ and $\partial V/\partial MDH$, respectively. The equations were solved with respect to $\partial V/\partial AAT$ and $\partial V/\partial MDH$, and we obtained equations in terms of $\mathbf{V}$, $NADH$ and enzyme concentrations, AAT and MDH, i.e.

$$\begin{aligned} der1 &= \partial V/\partial AAT = f_1(V, NADH, AAT, MDH) \\ der2 &= \partial V/\partial MDH = f_2(V, NADH, AAT, MDH) \end{aligned} \tag{6.2}$$

The Gröbner basis equation, F, was solved using NAG routine $C02AEF$ and the meaningful root was obtained. Using the meaningful root and values of $\partial V/\partial AAT$ and $\partial V/\partial MDH$ we calculated flux control coefficients at the end of $NADH$ time course

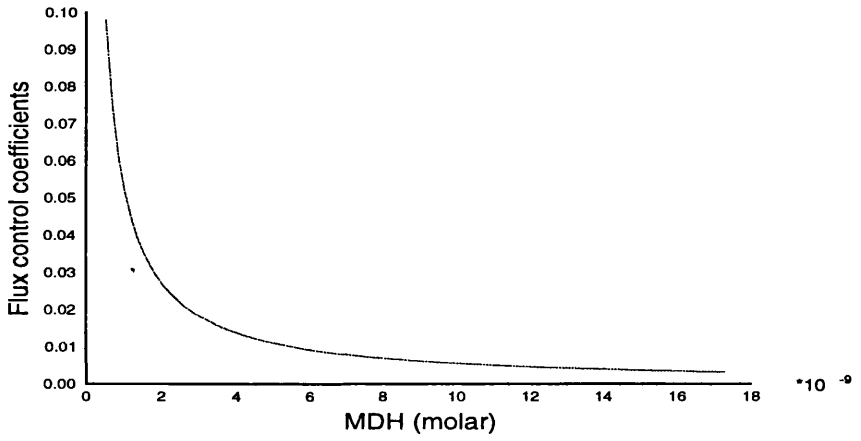


Figure 6-1: Graphical form of the flux control coefficient $C^{\mathbf{V}}_{MDH}$ against the enzyme concentration.

data concentration as follows:

$$C^{\mathbf{V}}_{AAT} = \frac{AAT}{\mathbf{V}} \times \frac{\partial \mathbf{V}}{\partial AAT} = 9.970 \times 10^{-1}$$

and

$$C^{\mathbf{V}}_{MDH} = \frac{MDH}{\mathbf{V}} \times \frac{\partial \mathbf{V}}{\partial MDH} = 2.252 \times 10^{-3}$$

As we see above, the magnitude of the $C^{\mathbf{V}}_{MDH}$ is 2.252×10^{-3}, which means that this enzyme has very little control and increases of its activity, either specific activity or concentration, are going to have little impact on the dependent flux and any character related to it.

6.1.1 The Summation Theorem

The flux control coefficient does not give any insight into metabolic control. All it states is how changes to enzyme activities of particular steps affect flux, and thereby gives information on how control is distributed amongst the different enzymes of the system. For further insight, one must turn to the theorems of control theory. One of them is the summation theorem. This states that the sum of all the flux control

coefficients of a reaction system is unity [Kacser and Burns, 1973]. Therefore,

$$C^{\mathbf{V}}_{AAT} + C^{\mathbf{V}}_{MDH} = 9.970 \times 10^{-1} + 2.252 \times 10^{-3} = 1.00$$

The summation theorem provides a linear constraint on the distribution of flux control coefficients. One consequence of this is that if there is a change in the steady state caused by a change in an external effector, then the distribution of flux control coefficients will readjust so that the summation theorem is obeyed. If some reactions experience a fall in their flux control coefficients, then the flux control coefficients for other steps must rise. The summation theorem also indicates that if there are any steps which have flux control coefficients greater than one, then there must be other steps which compensate by having negative flux control coefficients. The summation theorem puts constraints on the distribution of control such that if some enzymes have high control then others must have less. The summation theorem provides the notion that enzymes compete for the control of flux.

6.2 Concentration Control Coefficient

Concentration control coefficient describes how a change in an enzyme's activity affects the concentration of a metabolite in the system. A positive coefficient would indicate an increase in a metabolite level (for example, the product of an enzyme would rise in response to a rise in the enzyme's activity), whilst a negative coefficient would be the result of a fall in a metabolite concentration (for example, the substrate for an enzyme would fall if the enzyme's activity were increased). For the aspartate aminotransferase-malate dehydrogenase coupled system, concentration control coefficients are calculated using the following REDUCE format;

```
bas := groebner({ equ1, equ2, equ3, equ4, equ5, equ6, equ7 },
                { asp,  akg,  glu,  mal,  nad,  oaa,  v }) $
F    := num( part( bas, 7 )) $
fun1 := num( part( bas, 6 )) $
oaa  := part( solve ( fun1, oaa ), 1, 2 ) $
```

```
de1  := df( oaa, AAT ) $
de2  := df( oaa, MDH ) $
```

The equations $de1$ and $de2$ have the following form:

$$\begin{aligned} de1 &= \partial oaa/\partial AAT = g_1(V, NADH, AAT, MDH) \\ de2 &= \partial oaa/\partial MDH = g_2(V, NADH, AAT, MDH) \end{aligned} \tag{6.3}$$

The Gröbner basis equation F was solved using the NAG routine $C02AEF$ and meaningful root,$\mathbf{V}$, was obtained. Then the root was substituted into the derivative equations to obtain concentration control coefficients. Thus concentration control coefficients for the aspartate aminotransferase-malate dehydrogenase system at the end of time course data point are as follows:

$$C^{oaa}_{AAT} = \frac{AAT}{oaa} \times \frac{\partial oaa}{\partial AAT} = 9.686 \times 10^{-1}$$

and

$$C^{oaa}_{MDH} = \frac{MDH}{oaa} \times \frac{\partial oaa}{\partial MDH} = -9.686 \times 10^{-1}$$

There are as many concentration control coefficients for any one metabolite as there are enzymes and substrates in the system. The sum of all concentration control coefficients is always zero [Heinrich and Rapoport, 1974]. Thus,

$$C^{oaa}_{AAT} + C^{oaa}_{MDH} = 9.686 \times 10^{-1} - 9.686 \times 10^{-1} = 0.0000$$

This is known as the concentration summation property.

6.3 Elasticity Coefficient

The analysis presented so far has dealt with the effects on system properties such as flux control, or concentration control coefficients. The functional significance of each enzyme within the system may be assessed by considering it to be isolated from the rest of the system but surrounded by all its substrates and products. Let us consider the

response(δv_1) of the local reaction rate to a change (δoaa) in the concentration (oaa), all other metabolite concentrations being held constant, i.e, the elasticity, a local property, defines the fractional change that would occur in the enzyme's velocity due to a certain fractional change in the concentration of one of the substrates or products. To calculate an elasticity coefficient for aspartate aminotransferase and malate dehydrogenase, we eliminated all unwanted quantities using following REDUCE format:

```
bas := groebner({ equ1, equ2, equ3, equ4, equ5, equ6 },
                { asp,  akg,  glu,  mal,  nad,  v  }) $
fun1 := num( part( bas, 6 )) $
v1   := part( solve( fun1, v ), 1, 2 )$
%
bas1 := groebner({equ1, equ2, equ3, equ4, equ5, equ7 },
                 {asp,  akg,  glu,  mal,  nad,  v  }) $
fun2 := num( part( bas, 6 )) $
v2   := part( solve( fun2, v ), 1, 2 )$
```

Therefore we obtained the rate laws in terms of $NADH$ and oaa concentrations, i.e. of the form;

$$\begin{aligned} v_1 &= g_3(NADH, oaa) \\ v_2 &= g_4(NADH, oaa) \end{aligned} \tag{6.4}$$

The elasticity coefficients of aspartate aminotransferase and malate dehydrogenase with respect to oaa are calculated using the following REDUCE format:

```
elas(1) := df(v1, oaa)*(oaa/v1) $
elas(2) := df(v2, oaa)*(oaa/v1) $
```

The value of oaa is obtained using the steady state rate law (V) and substituted into the equations $elas(1)$ and $elas(2)$. Thus, the numerical value of the elasticity coefficients of the aspartate aminotransferase at the end of the time course data point are as follows:

$$\epsilon_{oaa}^{v_1} = \frac{oaa}{v_1} \times \frac{\partial v_1}{\partial oaa} = -2.326 \times 10^{-3}$$

Similarly the elasticity coefficient for malate dehydrogenase is:

$$\epsilon_{oaa}^{v_2} = \frac{oaa}{v_2} \times \frac{\partial v_2}{\partial oaa} = 1.030$$

By definition, the product of an elasticity coefficient (ϵ) and any small change in relevant metabolite concentration($\delta oaa/oaa$) gives the local response ($\delta v_1/v_1$), i.e. the resulting change in local reaction rate. The metabolite product should have a negative elasticity coefficient [Kacser and Porteous, 1987]. *oaa* is a product for the first reaction and its elasticity coefficient, $\epsilon_{oaa}^{v_1}$, is negative. The metabolite substrate in general has positive elasticity coefficient but at higher substrate concentrations negative coefficients may occur. *oaa* is a substrate for second reaction and its elasticity, $\epsilon_{oaa}^{v_2}$, is positive.

All elasticity coefficients are dimensionless numbers; their absolute value depends on the concentrations of all the interacting molecules and the kinetic constants of the enzyme involved in catalysing the local reaction velocity.

6.3.1 The Connectivity Theorem

So far, three coefficients of metabolic pathway have been defined: the flux control coefficient, concentration control coefficient and elasticity coefficient. The flux control coefficient is a system property and depends on the concerted action of all the enzymes and other parameters of the system; the elasticity in contrast is what is termed a local property and is the individual response of an isolated unit process, usually an enzyme. What, if any, is the connection between these two measures? If such a connection does exist then it would indicate for the first time how the intrinsic properties of the individual enzymes contribute to the properties of the whole system. The connectivity theorem as it is known, constitutes such a relationship [Kacser and Burns, 1973] and is given for the aspartate aminotransferase-malate dehydrogenase coupled system by

$$C_{AAT}^{V} \times \epsilon_{oaa}^{v1} + C_{MDH}^{V} \times \epsilon_{oaa}^{v2} = 0 \tag{6.5}$$

Given the connectivity theorem and the summation theorem, all the flux control coefficients for a linear chain can be written as a function of the elasticities only. This

indicates that the elasticities are of a fundamental nature, i.e, if elasticities are given, we can then find flux control coefficients and concentration control coefficients using connectivity and summation theorems.

6.3.2 The Response Coefficient

An altogether different relationship exists if responses to 'external' effectors are considered. Such external effectors are the (constant) initial and final pathway substrates, asp, α-kg, glu, NADH, NAD^+ and mal, which are external to the system. Their critical feature is that they are not variables which find their levels due to the interactions of the system but are fixed parameters. The responses to changes in these effectors are therefore not expressible in terms of the normal elasticities or control coefficients defined before.

Nevertheless, there are local and flux coefficients which define the action of such external effectors. The response of the effector *asp* on the enzyme rate is defined by the external elasticity coefficient.

$$\epsilon_{asp}^{v_1} = \frac{\partial v_1}{\partial asp} \cdot \frac{asp}{v_1} \tag{6.6}$$

the flux response coefficient for *asp* is defined by;

$$R_{\text{asp}}^{\text{V}} = \frac{\partial V}{\partial asp} \cdot \frac{asp}{V} \tag{6.7}$$

and the concentration response coefficients;

$$R_{\text{asp}}^{\text{oaa}} = \frac{\partial oaa}{\partial asp} \cdot \frac{asp}{oaa} \tag{6.8}$$

Kacser and Burns (1973) showed that the following relation exists between the response coefficient and the elasticities and flux control coefficients;

$$R_{\text{asp}}^{\text{V}} = C_{\text{E}_\text{i}}^{\text{V}} \cdot \epsilon_{asp}^{v_i} \tag{6.9}$$

or

$$C^{\mathrm{V}}_{\mathrm{E_i}} = \frac{R^{\mathrm{V}}_{\mathrm{asp}}}{\epsilon^{v_i}_{asp}} \tag{6.10}$$

This is the *combined response* relationship. Similarly

$$R^{\mathrm{oaa}}_{\mathrm{asp}} = C^{\mathrm{oaa}}_{\mathrm{E_i}} \cdot \epsilon^{v_i}_{asp} \tag{6.11}$$

or

$$C^{\mathrm{oaa}}_{\mathrm{E_i}} = \frac{R^{\mathrm{oaa}}_{\mathrm{asp}}}{\epsilon^{v_i}_{asp}} \tag{6.12}$$

The response coefficient is directly proportional to the flux control coefficient of the enzyme it affects, and the elasticity coefficient of the enzyme with respect to the external controller. The $\epsilon^{v_i}_{asp}$ notation on the elasticity, shows that, although operationally equivalent to the previously described elasticities, the elasticity, $\epsilon^{v_i}_{asp}$, is different in that it does not occur in the connectivity relationships for the system [Hofmeyr et al, 1986]. The equation,

$$R^{\mathrm{V}}_{\mathrm{asp}} = C^{\mathrm{V}}_{\mathrm{E_i}} \cdot \epsilon^{v_i}_{asp} \tag{6.13}$$

provides two important insights into the control of metabolic pathways:

1. the ability of an external controller to affect flux depends on a local property and a system property, studying the action of effectors on an isolated enzyme is not enough to infer the possibility of control by that effector;

2. an external effector's ability to control can be compensated by either a high elasticity or high flux control coefficient if the other coefficient is lacking in response. The classical approach to defining control points is therefore misleading because even if an enzyme has a poor ability to control flux, provided that the enzyme has a high elasticity with respect to the effector, the response of the flux to the external effector will be high. Nevertheless, the limited range of effector elasticities will limit the scale of this compensation.

Computer algebra system allows us to define such response coefficients analytically. Let us consider the aspartate aminotransferase-malate dehydrogenase coupled system.

The system has six external effectors (*asp*, α-kg, *glu*, $NADH$, NAD^+ and *mal*). and the system is chracterized by the steady state rate law $\mathbf{V} = v_1 = v_2$. The rate law v_1 is function of *asp*, α-kg, *glu*, number of kinetic parameters and Enzyme (AAT), i.e. it is in following form;

$$v_1 = f(asp, \alpha\text{-kg}, glu, oaa, k_i, AAT) \tag{6.14}$$

Similarly,

$$v_2 = f(NADH, oaa, NAD^+, mal, k_i, MADH) \tag{6.15}$$

The response coefficients for local rate laws v_1 and v_2 are calculated using the following REDUCE format:

```
% Calculation of response coefficients for v1.
            r1 := df( v1, asp )*( asp/v1 ) $
            r2 := df( v1, akg )*( akg/v1 ) $
            r3 := df( v1, glu )*( glu/v1 ) $
% Calculation of response coefficients for v2.
            r4 := df( v2, NADH )*( NADH/v2 ) $
            r5 := df( v2, NAD )*( NAD/v2 ) $
            r6 := df( v2, mal )*( mal/v2 ) $
```

Now we substitute the value of *asp*, *akg*, *glu*, NAD and *mal* after calculating in REDUCE by

```
 bas := groebner({ equ1, equ2, equ3, equ4, equ5},
                 { asp,  akg,  glu,  mal,  nad }) $
if length coeff(part(bas,5),nad) = 2 then
   nad := part(solve(num(part(bas,5)), nad),1,2 ) $
if length coeff(part(bas,4),mal) = 2 then
   mal := part(solve(num(part(bas,4)), mal),1,2 ) $
if length coeff(part(bas,3),glu) = 2 then
   glu := part(solve(num(part(bas,3)), glu),1,2 ) $
if length coeff(part(bas,2, akg) = 2 then
```

Coefficients	Theoretical val	Coefficients	Theoretical val
$\epsilon^{v_1}_{\text{asp}}$	1.191×10^{-3}	$\epsilon^{v_2}_{\text{NADH}}$	1.329×10^{-1}
$\epsilon^{v_1}_{\alpha\text{-kg}}$	9.991×10^{-1}	$\epsilon^{v_2}_{\text{mal}}$	-3.868×10^{-2}
$\epsilon^{v_1}_{\text{glu}}$	-1.163×10^{-3}	$\epsilon^{v_2}_{\text{NAD+}}$	-4.054×10^{-2}

Table 6.1: External elasticity coefficients for the aspartate aminotransferase and malate dehydrogenase isolated systems.

Coefficients	Theoretical val	Coefficients	Theoretical val
$R^{\text{V}}_{\text{asp}}$	1.188×10^{-3}	$R^{\text{V}}_{\text{NADH}}$	2.996×10^{-4}
$R^{\text{V}}_{\alpha\text{-kg}}$	9.969×10^{-1}	$R^{\text{V}}_{\text{mal}}$	-8.704×10^{-5}
$R^{\text{V}}_{\text{glu}}$	-1.160×10^{-3}	$R^{\text{V}}_{\text{NAD+}}$	-9.137×10^{-5}

Table 6.2: Flux response coefficients for the aspartate aminotransferase and malate dehydrogenase coupled system.

```
    akg := part(solve(num(part(bas,2)), akg),1,2 ) $
if length coeff(part(bas,1),asp) = 2 then
    asp := part(solve(num(part(bas,1)), asp),1,2 ) $
```

Therefore the response coefficients of the aspartate aminotransferase and malate dehydrogenase systems at the end of the time course data are as in tables 6.1, 6.2 and 6.3.

6.3.3 Fitting the Control Coefficients to Experimental Data

Since the discovery of enzymes, there has been a considerable effort by biochemists to isolate them and characterise their structural and kinetic properties. It would seem

Coefficients	Theoretical val	Coefficients	Theoretical val
$R^{\text{oaa}}_{\text{asp}}$	1.154×10^{-3}	$R^{\text{oaa}}_{\text{NADH}}$	-1.287×10^{-1}
$R^{\text{oaa}}_{\alpha\text{-kg}}$	9.678×10^{-1}	$R^{\text{oaa}}_{\text{mal}}$	3.741×10^{-2}
$R^{\text{oaa}}_{\text{glu}}$	-1.127×10^{-3}	$R^{\text{oaa}}_{\text{NAD+}}$	3.927×10^{-2}

Table 6.3: Concentration response coefficients for aspartate aminotransferase and malate dehydrogenase coupled system.

reasonable, therefore, to fit the control coefficients to experimental data.

Our rate law relates $\mathbf{V}$(i.e, $-d[NADH]/dt$ to $NADH$, while our experimental data is for $NADH$ and t. The rate law is of the form,

$$\mathbf{V} = -d[NADH]/dt = f(NADH, k_i, E_i) \tag{6.16}$$

and metabolite control coefficients are as follow,

$$\begin{aligned} \epsilon_{\text{oaa}}^{v_i} &= y_1(NADH, k_i, E_i) \\ C_{E_i}^{V} &= y_2(NADH, k_i, E_i) \\ R_{\text{ex}}^{\text{oaa}} &= y_3(NADH, k_i, E_i) \\ C_{E_i}^{\text{oaa}} &= y_4(NADH, k_i, E_i) \end{aligned} \tag{6.17}$$

where ex is asp, akg, glu, mal, $NADH$ and NAD. We need to integrate the rate law to fit kinetic constants to experimental data. For this process, Rate law and first implicit derivatives of $\mathbf{V}$ with respect to each enzyme were calculated in REDUCE and their FORTRAN expression were produced using GENTRAN. The rate law was integrated using a NAG routine to fit kinetic parameters to experimental data. This has been done in the previous section(fitting the kinetic parameters to experimental data). After fitting the kinetic parameters to experimental data, we substituted them in the formulae for the control coefficients and obtained estimation of metabolite control coefficients. This was achieved using scaled kinetic constants and their result was showed in tables 6.4, 6.5, 6.6 and 6.7. This shows how the control coefficients depend on the individual kinetic properties of the system.

6.3.4 summary

From the definition of metabolite and flux control coefficients and the elasticities we are able to derive symbolic forms of these functions, in terms of conventional kinetic parameters. At the simplest level we are able to substitute values of these kinetic parameters, to yield values for the metabolic control coefficients. Since we are substituting into symbolic equations we can always guarantee the conservation

Coefficients	Theoretical value	Experimental value
$\epsilon^{v_1}_{\text{oaa}}$	-2.326×10^{-3}	$-2.799 \pm 0.031 \times 10^{-3}$
$\epsilon^{v_2}_{\text{oaa}}$	1.030×10^{-0}	$1.031 \pm 0.001 \times 10^{-0}$
$C^{\text{V}}_{\text{AAT}}$	9.970×10^{-1}	$9.973 \pm 0.002 \times 10^{-1}$
$C^{\text{V}}_{\text{MDH}}$	2.252×10^{-3}	$2.709 \pm 0.030 \times 10^{-3}$
$C^{\text{oaa}}_{\text{AAT}}$	9.686×10^{-1}	$9.677 \pm 0.023 \times 10^{-1}$
$C^{\text{oaa}}_{\text{MDH}}$	-9.686×10^{-1}	$-9.677 \pm 0.023 \times 10^{-1}$

Table 6.4: Fitting of the elasticity, flux control and concentration control coefficients to the experimental data

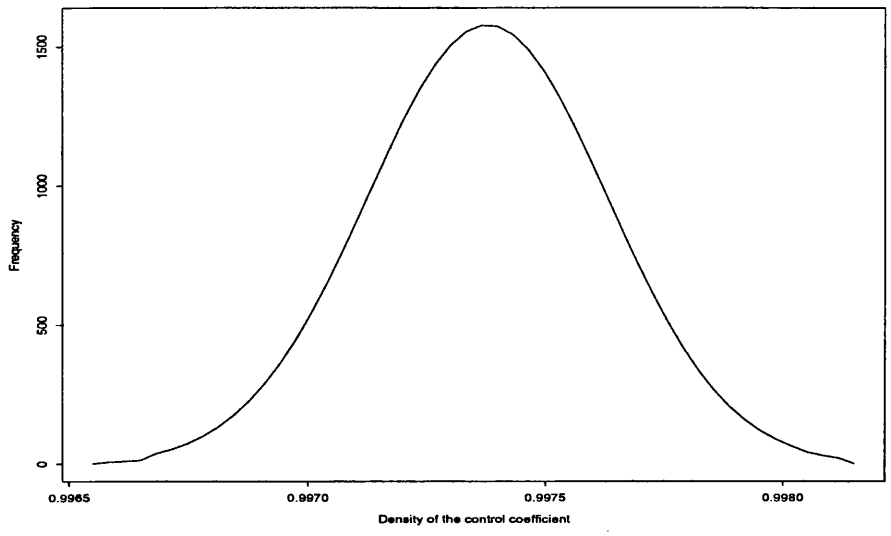


Figure 6-2: A statistical distribution of the flux control coefficient C^V_{AAT}.

Coefficients	Theoretical value	Experimental value
$\epsilon^{v_1}_{\text{asp}}$	1.191×10^{-3}	$1.565 \pm 0.024 \times 10^{-3}$
$\epsilon^{v_1}_{\alpha\text{-kg}}$	9.991×10^{-1}	$9.999 \pm 0.001 \times 10^{-1}$
$\epsilon^{v_1}_{\text{glu}}$	-1.163×10^{-3}	$-1.539 \pm 0.024 \times 10^{-3}$
$\epsilon^{v_2}_{\text{NADH}}$	1.329×10^{-1}	$1.284 \pm 0.003 \times 10^{-1}$
$\epsilon^{v_2}_{\text{mal}}$	-3.868×10^{-2}	$-3.903 \pm 0.001 \times 10^{-2}$
$\epsilon^{v_2}_{\text{NAD+}}$	-4.054×10^{-2}	$-4.052 \pm 0.002 \times 10^{-2}$

Table 6.5: Fitting of the external elasticity coefficients to the experimental data.

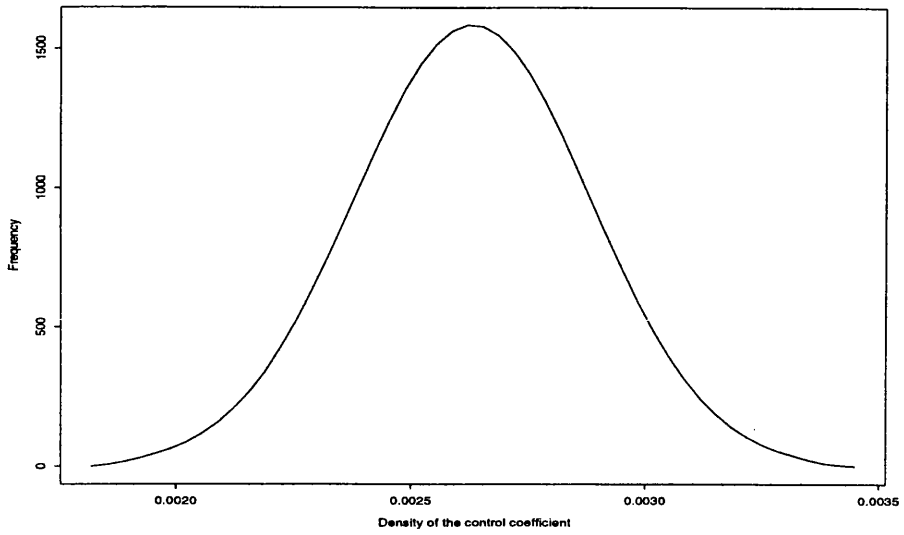


Figure 6-3: A statistical distribution of the flux control coefficient C^V_{MDH}.

Coefficients	Theoretical value	Experimental value
$R^{\mathrm{V}}_{\mathrm{asp}}$	1.188×10^{-3}	$1.561 \pm 0.024 \times 10^{-3}$
$R^{\mathrm{V}}_{\alpha\text{-kg}}$	9.969×10^{-1}	$9.972 \pm 0.001 \times 10^{-1}$
$R^{\mathrm{V}}_{\mathrm{glu}}$	-1.160×10^{-3}	$-1.535 \pm 0.018 \times 10^{-3}$
$R^{\mathrm{V}}_{\mathrm{NADH}}$	2.996×10^{-4}	$3.479 \pm 0.030 \times 10^{-4}$
$R^{\mathrm{V}}_{\mathrm{mal}}$	-8.704×10^{-5}	$-1.057 \pm 0.012 \times 10^{-4}$
$R^{\mathrm{V}}_{\mathrm{NAD+}}$	-9.137×10^{-5}	$-1.o98 \pm 0.011 \times 10^{-4}$

Table 6.6: Fitting of the flux response coefficients to the experimental data.

Coefficients	Theoretical val	Experimental val.
$R^{\mathrm{oaa}}_{\mathrm{asp}}$	1.154×10^{-3}	$1.515 \pm 0.023 \times 10^{-3}$
$R^{\mathrm{oaa}}_{\alpha\text{-kg}}$	9.678×10^{-1}	$9.677 \pm 0.001 \times 10^{-1}$
$R^{\mathrm{oaa}}_{\mathrm{glu}}$	-1.127×10^{-3}	$-1.489 \pm 0.024 \times 10^{-3}$
$R^{\mathrm{oaa}}_{\mathrm{NADH}}$	-1.287×10^{-1}	$-1.243 \pm 0.002 \times 10^{-1}$
$R^{\mathrm{oaa}}_{\mathrm{mal}}$	3.741×10^{-2}	$3.777 \pm 0.001 \times 10^{-2}$
$R^{\mathrm{oaa}}_{\mathrm{NAD+}}$	3.927×10^{-2}	$3.921 \pm 0.002 \times 10^{-2}$

Table 6.7: Fitting of concentration response coefficient to the experimental data.

relationships hold.

We have done this in this chapter for two enzyme system *in vitro* using aspartate aminotransferase malate dehydrogenase. This is a major advance on existing measurement methodologies, which rely on genetically manipulating the enzyme concentrations to determine these coefficients, with considerable experimental error.

Chapter 7

Automatic Derivation of Steady State Rate Laws

The derivation of a rate law according to steady-state assumption for a particular enzyme mechanism essentially involves the solution of a set of simultaneous non-homogeneous linear equations. These may be solved in straightforward fashion using Cramer's method [Kistiakowsky and Shaw, 1953], but unless the mechanism is simple, this is an extremely laborious technique.

An alternative, relatively simple, schematic procedure for deriving steady-state rate laws was first described over thirty years ago [King and Altman, 1956]. This procedure has been computerised, although with some difficulty [Fisher and Schultz, 1969]. More recently improved algorithms for derivation of rate laws using this approach have been suggested [Lam and Priest, 1972; Lam and Schutz, 1978].

We suggest that the original formulation as a set of simultaneous equations is now more appropriate for computer generation of steady-state rate laws. With a computer algebra system and the new mathematical technique of Gröbner Bases [Buchberger, 1985] we show how a rate law may be derived. The technique allows us to express the rate law in whatever kinetic parameters are convenient—Michaelis constants, K_M and V_{max}'s or the individual rate constants of the steps of the mechanism.

7.1 Computing Method to Derive the Steady State Rate Laws

To obtain the rate law for a specific mechanism one has to first define the differential equations and the enzyme conservation equation that describe the mechanism. The differential equations are written in terms of the kinetic constants of the elementary reactions of the mechanism k_i and k_{-i}. Assuming a steady-state these form a system of ordinary equations which is solved using the mathematical technique of Gröbner Bases [Buchberger, 1985]. The result is the rate equation expressed in terms of the k_i's. Rate equations in this form may not be very useful if they are going to be used as models for experimental enzyme kinetics. In that case one would like to have these equations expressed with more convenient parameters such as K_M or V_{max}'s. There is, however, more than one set of such parameters in multisubstrate multiproduct reactions due to the appearance of crossed terms such as AB, PQR, etc. Several authors have used different sets of parameters [Alberty, 1953; Dalziel, 1957; Cleland, 1963 and Bloomfield *et al*, 1962] and choosing a suitable one should be left to personal preferences.

The program here described starts by calculating the rate laws in term of individual kinetic parameters which are the easiest to get. All the other sets of parameters such as Alberty's parameters can then be obtained from these. Our method is purely algorithmic to derive the steady state rate laws. The method is as follows as a algorithm:

1. Write down the rate of formation of all intermediates, and making the quasi-steady state assumption equate these to zero.

2. Write down the conservation relationships for all enzymes and intermediates.

3. Solve these simultaneous linear equations, to eliminate all intermediates, and obtain a rate law in terms of simple rate constants.

4. If expressions for quantities such as V_{max} and K_M are desired, derive these by substitution from their definition.

To follow this algorithm by hand for complex mechanisms is too difficult, and so

graphical techniques, such as the King-Altman procedure have been devised [King and Altman, 1956]. Such techniques are not ideally suited to computerisation.

The use of computer algebra makes our obvious algorithm feasible. As a practical example the rate laws of the aspartate aminotransferase and malate dehydrogenase were both derived from their mechanism, and shown to be identical to the published rate laws.

7.1.1 Derivation of Two Substrate Reaction Rate-Laws

One can always represent the rate law for an irreversible reaction (or a reversible one with products or substrates all set to zero) in the Linewaever-burk form. For simplicity we begin by considering a two substrate and two product reaction such as

$$\mathrm{A} + \mathrm{B} \quad \rightleftharpoons \quad \mathrm{P} + \mathrm{Q}.$$

Therefore, the forward and reverse reaction rate-laws are then:

$$\frac{1}{v} = \frac{1}{V^F_{max}} \cdot (1 + \frac{K_A}{[A]} + \frac{K_B}{[B]} + \frac{K_{AB}}{[A][B]}). \tag{7.1}$$

and

$$\frac{1}{v} = \frac{1}{V^R_{max}} \cdot (1 + \frac{K_P}{[P]} + \frac{K_Q}{[Q]} + \frac{K_{PQ}}{[P][Q]}). \tag{7.2}$$

The method can be generalised to more enzyme systems, for example, a n-substrate and n-product reaction such as

$$X_o + X_1 + \cdots + X_n \quad \rightleftharpoons \quad Y_o + Y_1 + \cdots + Y_n.$$

Therefore, we can write the forward reaction rate-law as follow:

$$\frac{1}{v} = \frac{1}{V^F_{max}} \cdot (1 + \frac{K_{X_o}}{[X_o]} + \frac{K_{X_1}}{[X_1]} + \cdots + \frac{K_{X_oX_1}}{[X_o][X_1]} + \cdots + \frac{K_{X_oX_1\cdots X_n}}{[X_o][X_1]\cdots[X_n]}). \tag{7.3}$$

An equivalent rate law for the reverse direction can be written.

We here calculate Alberty's parameters which are the easiest to get. A short description of these parameters:

1. Catalytic constants.

   - $V^F_{max} = k^F_{cat} \cdot E_0$—maximum velocity of forward reaction.
   - $V^R_{max} = k^R_{cat} \cdot E_0$—maximum velocity of reverse reaction.

   V_{max} have the same meaning of Henri-Michaelis-Menten V_{max}: the saturating velocities.

2. Specificity constants.

   - K_A—Apparent Michaelis constant for substrate A, i.e. concentration of A for which $v = V^F_{max}/2$ with all other substrates saturating and zero concentration of products.
   - K_P—Apparent Michaelis constant for product P, i.e. concentration of P for which $v = V^R_{max}/2$ with all other products saturating and zero concentration of substrates.
   - K_{AB}—joint specificity constant for substrates A and B.
   - K_{PQ}—joint specificity constant for products P and Q.

   In the case of reactions with more than two substrates or products, several other joint specificity constants will appear such as $K_{X_0X_1X_2}$.

The joint specificity constants do not have an easily understandable physical meaning and this is the reason for Cleland [Cleland, 1963] having used a different set of parameters. However, the Alberty set is easier to handle algebraically.

Although some of the crossed terms can disappear for some specific mechanisms, the rate laws of mechanisms with modifiers will obviously have more terms, which can still be obtained by this procedure.

To execute the procedures described above an example computer program was written to be executed under REDUCE which is a program for manipulation of algebraic expressions, and given in appendix A.

The program was used to determine the rate equation of the malate dehydrogenase mechanism. This rate equation is shown below in three different form.

1. Using individual rate constants (More details were given chapter 4).

$$
\begin{aligned}
v \;=\; & ([E_0]\cdot(k_1\cdot[NADH]\cdot k_3\cdot[oaa]\cdot k_5\cdot k_7 - k_4\cdot k_2\cdot k_6\cdot[mal]\cdot k_8\cdot \\
& [NAD]))/(k_1\cdot[NADH]\cdot k_4\cdot k_6\cdot[mal] + k_1\cdot[NADH]\cdot k_4\cdot k_7 \\
& + k_1\cdot[NADH]\cdot k_3\cdot[oaa]\cdot k_6\cdot[mal] + k_1\cdot[NADH]\cdot k_3\cdot[oaa]\cdot k_5 \\
& + k_1\cdot[NADH]\cdot k_3\cdot[oaa]\cdot k_7 + k_1\cdot[NADH]\cdot k_5\cdot k_7 + k_4\cdot k_2\cdot k_6\cdot \\
& [mal] + k_4\cdot k_2\cdot k_8\cdot[NAD] + k_4\cdot k_2\cdot k_7 + k_4\cdot k_6\cdot[mal]\cdot k_8\cdot \\
& [NAD] + k_2\cdot k_6\cdot[mal]\cdot k_8\cdot[NAD] + k_2\cdot k_5\cdot k_8\cdot[NAD] + k_2\cdot \\
& k_5\cdot k_7 + k_3\cdot[oaa]\cdot k_6\cdot[mal]\cdot k_8\cdot[NAD] + k_3\cdot[oaa]\cdot k_5\cdot k_8\cdot \\
& [NAD] + k_3\cdot[oaa]\cdot k_5\cdot k_7)
\end{aligned}
\tag{7.4}
$$

2. Using Alberty's parameters:

We have written forward and reverse reaction rate-laws, like 7.1 and 7.2, which were of the forms:

$$
\begin{aligned}
f_f &= f_1(V^{\mathrm{F}}_{\mathrm{max}}, K^{\mathrm{oaa}}_{\mathrm{M}}, K^{\mathrm{NADH}}_{\mathrm{M}}, K^{\mathrm{oaa.NADH}}_{\mathrm{M}}, oaa, NADH) \\
f_r &= f_2(V^{\mathrm{R}}_{\mathrm{max}}, K^{\mathrm{NAD^+}}_{\mathrm{M}}, K^{\mathrm{mal}}_{\mathrm{M}}, K^{\mathrm{mal.NAD^+}}_{\mathrm{M}}, mal, NAD)
\end{aligned}
\tag{7.5}
$$

and the rate-law in terms of individual rate constants which was obtained before (the equation 7.4),

$$
rate_{ind} = v = g_1(E_o, k_i, oaa, NADH, NAD, mal). \tag{7.6}
$$

We can now write an equivalent rate law for the forward and reverse direction. First of all we obtained the inverses of the computed rate law $rate_{ind}$.

$$
\begin{aligned}
rate_{f_{inv}} &= 1/sub(mal = 0, nad = 0, rate_{ind}) \\
rate_{r_{inv}} &= 1/sub(oaa = 0, nadh = 0, rate_{ind})
\end{aligned}
\tag{7.7}
$$

Now $rate_{f_{inv}}$ holds for all concentrations of oaa and $NADH$, and $rate_{r_{inv}}$ for all concentrations of mal and NAD, whether we express the rate law in terms of Alberty's parameters or in terms of individual rate constants. Thus the coefficients

of the metabolites must be identical however expressed. We thus write

$$rate_{f_{inv}} = f_f \tag{7.8}$$

and equate powers of the metabolites on each side. For convenience we remove denominators, so we in fact equate powers for

$$num(rate_{f_{inv}}) \times den(f_f) = num(f_f) \times den(rate_{f_{inv}}) \tag{7.9}$$

Therefore,

$$\begin{aligned} equ_1 &= num(rate_{f_{inv}}) \times den(f_f) \\ equ_2 &= num(f_f) \times den(rate_{f_{inv}}) \\ equ_3 &= num(rate_{r_{inv}}) \times den(f_r) \\ equ_4 &= num(f_r) \times den(rate_{r_{inv}}) \end{aligned} \tag{7.10}$$

All of these are quadratic in the coefficients, so we equate first and second degree. This gives us 8 equations which can be written using following REDUCE format:

```
s1  := coeffn( coeffn( eq1, nadh,1), oaa,1) -
       coeffn( coeffn( eq2, nadh,1), oaa,1) $
s2  := coeffn( coeffn( eq1, nadh,1), oaa,2) -
       coeffn( coeffn( eq2, nadh,1), oaa,2) $
s3  := coeffn( coeffn( eq1, nadh,2), oaa,1) -
       coeffn( coeffn( eq2, nadh,2), oaa,1) $
s4  := coeffn( coeffn( eq1, nadh,2), oaa,2) -
       coeffn( coeffn( eq2, nadh,2), oaa,2) $
s5  := coeffn( coeffn( eq3, nad,1), mal,1) -
       coeffn( coeffn( eq4, nad,1), mal,1) $
s6  := coeffn( coeffn( eq3, nad,1), mal,2) -
       coeffn( coeffn( eq4, nad,1), mal,2) $
s7  := coeffn( coeffn( eq3, nad,2), mal,1) -
       coeffn( coeffn( eq4, nad,2), mal,1) $
```

```
s8  := coeffn( coeffn( eq3, nad,2), mal,2) -
       coeffn( coeffn( eq4, nad,2), mal,2) $
```

Now we can solve these equations using Gröbner basis technique, **groesolve.**

```
bas :=groesolve({ s1, s2, s3, s4, s5, s6, s7, s8 },
                { k1, k2, k3, k4, k5, k6, k7, k8 }) $
bas1 := part( bas, 1 ) $
%
k1 := rhs( part( bas1, 1 ) $
k2 := rhs( part( bas1, 2 ) $
k3 := rhs( part( bas1, 3 ) $
k4 := rhs( part( bas1, 4 ) $
k5 := rhs( part( bas1, 5 ) $
k6 := rhs( part( bas1, 6 ) $
k7 := rhs( part( bas1, 7 ) $
k8 := rhs( part( bas1, 8 ) $
%
```

The value of k_i is substituted into the rate law $\mathbf{rate_{ind}}$, and we obtained the rate law in terms of Alberty's parameters such as

$$v = \frac{\frac{V^{F}_{max}[NADH][oaa]}{K_M^{oaa.NADH}} - \frac{V^{R}_{max}[mal][NAD^+]}{K_M^{mal.NAD^+}}}{\begin{array}{l} 1 + \frac{[NADH]K_M^{oaa}}{K_M^{oaa.NADH}}\frac{K_M^{NADH}[oaa]}{K_M^{oaa.NADH}} + \frac{K_M^{NAD^+}[mal]}{K_M^{mal.NAD^+}} + \frac{[NAD^+]K_M^{mal}}{K_M^{mal.NAD^+}} \\ + \frac{[NADH][oaa]}{K_M^{oaa.NADH}} + \frac{K_M^{NAD^+}K_M^{oaa}[NADH][mal]}{K_M^{oaa.NADH}K_M^{mal.NAD^+}} + \frac{K_M^{NADH}K_M^{mal}[oaa][NAD^+]}{K_M^{oaa.NADH}K_M^{mal.NAD^+}} \\ + \frac{[mal][NAD^+]}{K_M^{mal.NAD^+}} + \frac{[NADH][oaa][mal]}{K_M^{mal.NADH.oaa}} + \frac{[oaa][mal][NAD^+]}{K_M^{mal.oaa.NAD^+}} \end{array}} \tag{7.11}$$

where

$$K_{\mathrm{M}}^{\mathrm{mal.oaa.NAD^+}} = \frac{K_{\mathrm{M}}^{\mathrm{mal.NAD^+}\,2} K_{\mathrm{M}}^{\mathrm{oaa.NADH}^2} V_{\mathrm{max}}^{\mathrm{F}}}{\begin{aligned} &K_{\mathrm{M}}^{\mathrm{NADH}} K_{\mathrm{M}}^{\mathrm{mal.NAD^+}} K_{\mathrm{M}}^{\mathrm{oaa.NADH}} V_{\mathrm{max}}^{\mathrm{R}} \\ &+K_{\mathrm{M}}^{\mathrm{NADH}} K_{\mathrm{M}}^{\mathrm{mal.NAD^+}} K_{\mathrm{M}}^{\mathrm{oaa.NADH}} V_{\mathrm{max}}^{\mathrm{F}} \\ &-K_{\mathrm{M}}^{\mathrm{NADH}} K_{\mathrm{M}}^{\mathrm{mal.NAD^+}} K_{\mathrm{M}}^{\mathrm{NADH}} K_{\mathrm{M}}^{\mathrm{oaa}} V_{\mathrm{max}}^{\mathrm{F}} \\ &-K_{\mathrm{M}}^{\mathrm{NADH}} K_{\mathrm{M}}^{\mathrm{mal}} K_{\mathrm{M}}^{\mathrm{NAD^+}} K_{\mathrm{M}}^{\mathrm{oaa.NADH}} V_{\mathrm{max}}^{\mathrm{R}} \end{aligned}} \tag{7.12}$$

$$K_{\mathrm{M}}^{\mathrm{mal.NADH.oaa}} = \frac{K_{\mathrm{M}}^{\mathrm{mal.NAD^+}\,2} K_{\mathrm{M}}^{\mathrm{oaa.NADH}^2} V_{\mathrm{max}}^{\mathrm{F}}}{\begin{aligned} &K_{\mathrm{M}}^{\mathrm{NAD^+}} K_{\mathrm{M}}^{\mathrm{mal.NAD^+}} K_{\mathrm{M}}^{\mathrm{oaa.NADH}} V_{\mathrm{max}}^{\mathrm{R}} \\ &+K_{\mathrm{M}}^{\mathrm{NAD^+}} K_{\mathrm{M}}^{\mathrm{mal.NAD^+}} K_{\mathrm{M}}^{\mathrm{oaa.NADH}} V_{\mathrm{max}}^{\mathrm{F}} \\ &-K_{\mathrm{M}}^{\mathrm{NAD^+}} K_{\mathrm{M}}^{\mathrm{mal.NAD^+}} K_{\mathrm{M}}^{\mathrm{NADH}} K_{\mathrm{M}}^{\mathrm{oaa}} V_{\mathrm{max}}^{\mathrm{F}} \\ &-K_{\mathrm{M}}^{\mathrm{NAD^+}} K_{\mathrm{M}}^{\mathrm{mal}} K_{\mathrm{M}}^{\mathrm{NAD^+}} K_{\mathrm{M}}^{\mathrm{oaa.NADH}} V_{\mathrm{max}}^{\mathrm{R}} \end{aligned}} \tag{7.13}$$

Relationships between Alberty's parameters and rate constants as follows;

$$\begin{aligned}
V_{max}^{F} &= (k_5 \cdot k_7 \cdot [E_0])/(k_5 + k_7) \\
V_{max}^{R} &= (k_2 \cdot k_4 \cdot [E_0])/(k_2 + k_4) \\
K_{\mathrm{M}}^{\mathrm{NADH}} &= (k_5 \cdot k_7)/(k_1 \cdot (k_5 + k_7)) \\
K_{\mathrm{M}}^{\mathrm{NAD^+}} &= (k_2 \cdot k_4)/((k_2 + k_4) \cdot k_8) \\
K_{\mathrm{M}}^{\mathrm{oaa}} &= ((k_4 + k_5) \cdot k_7)/(k_3 \cdot (k_5 + k_7)) \\
K_{\mathrm{M}}^{\mathrm{mal}} &= (k_2 \cdot (k_4 + k_5))/((k_2 + k_4) \cdot k_6) \\
K_{\mathrm{M}}^{\mathrm{mal.NAD^+}} &= (k_2.k_7(k_4 + k_5))/(k_6.k_8(k_2 + k_4)) \\
K_{\mathrm{M}}^{\mathrm{oaa.NADH}} &= ((k_2.k_7(k_4 + k_5))/(k_1.k_3(k_5 k_7) \\
K_{\mathrm{M}}^{\mathrm{mal.oaa.NAD^+}} &= ((k_2.k_7(k_4 + k_5))/(k_6.k_8.k_3) \\
K_{\mathrm{M}}^{\mathrm{mal.NADH.oaa}} &= ((k_2.k_7(k_4 + k_5))/(k_1.k_3.k_6)
\end{aligned} \tag{7.14}$$

3. Using K_M, V_{max} and K_I.

$$v = \frac{\frac{V_{\mathrm{max}}^{\mathrm{F}}[\mathrm{NADH}][\mathrm{oaa}]}{K_{\mathrm{I}}^{\mathrm{NADH}} \cdot K_{\mathrm{M}}^{\mathrm{oaa}}} - \frac{V_{\mathrm{max}}^{\mathrm{R}}[\mathrm{mal}][\mathrm{NAD^+}]}{K_{\mathrm{M}}^{\mathrm{mal}} \cdot K_{\mathrm{I}}^{\mathrm{NAD^+}}}}{\begin{aligned} &1 + \frac{[\mathrm{NADH}]}{K_{\mathrm{I}}^{\mathrm{NADH}}} + \frac{K_{\mathrm{M}}^{\mathrm{NADH}}[\mathrm{oaa}]}{K_{\mathrm{I}}^{\mathrm{NADH}} \cdot K_{\mathrm{M}}^{\mathrm{oaa}}} + \frac{K_{\mathrm{M}}^{\mathrm{NAD^+}}[\mathrm{mal}]}{K_{\mathrm{M}}^{\mathrm{mal}} \cdot K_{\mathrm{I}}^{\mathrm{NAD^+}}} + \frac{[\mathrm{NAD^+}]}{K_{\mathrm{I}}^{\mathrm{NAD^+}}} \\ &+ \frac{[\mathrm{NADH}][\mathrm{oaa}]}{K_{\mathrm{I}}^{\mathrm{NADH}} \cdot K_{\mathrm{M}}^{\mathrm{oaa}}} + \frac{K_{\mathrm{M}}^{\mathrm{NAD^+}}[\mathrm{NADH}][\mathrm{mal}]}{K_{\mathrm{I}}^{\mathrm{NADH}} \cdot K_{\mathrm{M}}^{\mathrm{mal}} \cdot K_{\mathrm{I}}^{\mathrm{NAD^+}}} + \frac{K_{\mathrm{M}}^{\mathrm{NADH}}[\mathrm{oaa}][\mathrm{NAD^+}]}{K_{\mathrm{I}}^{\mathrm{NADH}} \cdot K_{\mathrm{M}}^{\mathrm{oaa}} \cdot K_{\mathrm{I}}^{\mathrm{NAD^+}}} \\ &+ \frac{[\mathrm{mal}][\mathrm{NAD^+}]}{K_{\mathrm{M}}^{\mathrm{mal}} \cdot K_{\mathrm{I}}^{\mathrm{NAD^+}}} + \frac{[\mathrm{NADH}][\mathrm{oaa}][\mathrm{mal}]}{K_{\mathrm{I}}^{\mathrm{NADH}} \cdot K_{\mathrm{M}}^{\mathrm{oaa}} \cdot K_{\mathrm{I}}^{\mathrm{mal}}} + \frac{[\mathrm{oaa}][\mathrm{mal}][\mathrm{NAD^+}]}{K_{\mathrm{I}}^{\mathrm{oaa}} \cdot K_{\mathrm{M}}^{\mathrm{mal}} \cdot K_{\mathrm{I}}^{\mathrm{NAD^+}}} \end{aligned}} \tag{7.15}$$

This is equivalent to the rate law 7.11, by the following relationship:

$$\begin{aligned} K_{\mathrm{M}}^{\mathrm{mal.oaa.NAD^+}} &= K_{\mathrm{I}}^{\mathrm{oaa}}.K_{\mathrm{M}}^{\mathrm{mal}}.K_{\mathrm{I}}^{\mathrm{NAD^+}} \\ K_{\mathrm{M}}^{\mathrm{mal.NADH.oaa}} &= K_{\mathrm{I}}^{\mathrm{NADH}}.K_{\mathrm{M}}^{\mathrm{oaa}}.K_{\mathrm{I}}^{\mathrm{mal}} \\ K_{\mathrm{M}}^{\mathrm{mal.NAD^+}} &= K_{\mathrm{I}}^{\mathrm{NAD^+}}.K_{\mathrm{M}}^{\mathrm{mal}} \\ K_{\mathrm{M}}^{\mathrm{oaa.NADH}} &= K_{\mathrm{I}}^{\mathrm{NADH}}.K_{\mathrm{M}}^{\mathrm{oaa}} \end{aligned} \tag{7.16}$$

The rate laws 7.4, 7.11 and 7.15 are equivalent by given relationship.

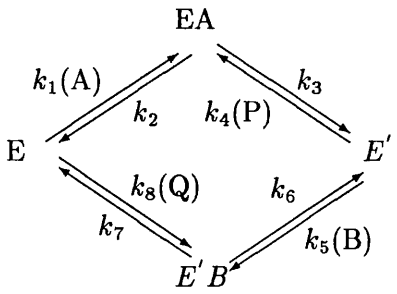
The ability of providing rate equations in several different representations is certainly of great use, especially when the objective is the experimental determination of the kinetic parameters. In this case, some sets of parameters are better for this purpose than others, but one cannot decide which a priori. Also, as relationships between the different representations are given, after determinating one set of parameters all other follow straightforward, even the rate constants of the elementary reaction steps.

7.1.2 An Example for Derivation Steady State Rate laws

The King and Altman procedure is based on matrix algebra and essentially simplifies the derivations by exploiting the inherent pattern to the full, and by using a shorthand notation [King and Altman, 1956]. Let us consider following mechanism,

$$\mathrm{E} \underset{k_2}{\overset{k_1(A)}{\rightleftharpoons}} \mathrm{EA} \underset{k_4(P)}{\overset{k_3}{\rightleftharpoons}} E' \underset{k_6}{\overset{k_5(B)}{\rightleftharpoons}} E'B \underset{k_8(Q)}{\overset{k_7}{\rightleftharpoons}} \mathrm{E}$$

The mechanism involves 4 enzyme-containing species; thus each term in the expression for the relative concentration of E, EA, $E'B$, or E' contains 4 rate constants and the appropriate concentration factors. First we write out the mechanism in a closed loop form so that all enzyme species, including E, occur only once:



We can set up steady-state equations in the usual way:

$$\begin{aligned} d[EA]/dt &= k_1.A.E + k_4.P.E' - (k_2 + k_3).EA = 0 \\ d[E']/dt &= k_3.EA + k_6.E'B - k_4.P.E' - k_5.E'.B = 0 \\ d[E'B]/dt &= k_5.B.E' + k_8.Q.E - (k_6 + k_7.E'B = 0 \end{aligned} \tag{7.17}$$

We now introduce the enzyme conservation equation:

$$E_o = E + E' + E'B + EA \tag{7.18}$$

The net rate in the forward direction is given by King-Altman:

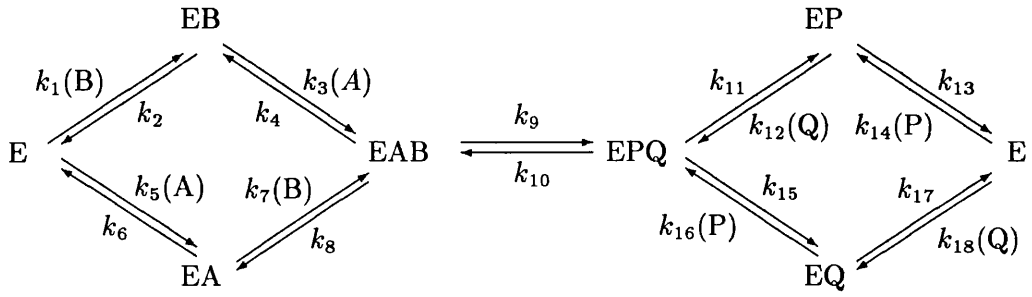
$$\frac{d[Q]}{dt} = v = \frac{(k_7.E'B - k_8.Q.E).E_o}{E+E'+E'B+EA} \tag{7.19}$$

If we solve the above 4 equations (7.17 and 7.18) with respect to 4 unknown, $E'B$, E', EA and E, and substitute them into the equation 7.19, we find following rate law;

$$\begin{aligned} v = {} & (E_o.([A].[B].k_1.k_3.k_5.k_7 - k_2.k_4.k_6.k_8.[P].[Q]))/ \\ & ([A].[B].k_1.k_3.k_5 + [A].[B].k_1.k_5.k_7 + [A].k_1.k_3.k_6 \\ & + [A].k_1.k_3.k_7 + [A].k_1.k_4.k_6.[P] + [A].k_1.k_4.k_7.[P] \\ & + [B].k_2.k_5.k_7 + [B].k_2.k_5.k_8.[Q] + [B].k_3.k_5.k_7 \\ & + [B].k_3.k_5.k_8.[Q] + k_2.k_4.k_6.[P] + k_2.k_4.k_7.[P] \\ & + k_2.k_4.k_8.[P].[Q] + k_2.k_6.k_8.[Q] + k_3.k_6.k_8.[Q] \\ & + k_4.k_6.k_8.[P].[Q]) \end{aligned} \tag{7.20}$$

This equation can be simplified to give the initial rate in the absence of products by setting $[P] = [Q] = 0$, or the initial rate in the reverse direction in the absence of substrates by setting $[A] = [B] = 0$.

So far, we have considered two substrate enzyme mechanism of which rate law can be obtained using King-Altman method. Let us now consider following enzyme mechanism. It is already pointed out that the derivation of full steady state rate law for this mechanism is extremely difficult [Cha, 1968; Wong and Hanes, 1962].



Computer algebra systems, however, can simplify the rate law. In terms of the rate constants shown, the steady-state equations for all the intermediates as follows;

$$\begin{aligned}
d[EB]/dt &= k_1.[B].[E] + k_4.[EAB] - (k_2 + k_3.[A]).[EB] \\
d[EA]/dt &= k_5.[A].[E] + k_8.[EAB] - (k_6 + k_7.[B]).[EA] \\
d[EAB]/dt &= k_3.[A].[EB] + k_7.[B].[EA] + k_{10}.[EPQ] - (k_4 + k_8 + k_9).[EAB] \\
d[EPQ]/dt &= k_9.[EAB] + k_{12}.[Q].[EP] + k_{16}.[P].[EQ] - (k_{10} + k_{11} + k_{15}).[EPQ] \\
d[EP]/dt &= k_{11}.[EPQ] + k_{14}.[P].[E] - (k_{12}.[Q] + k_{13}).[EP] \\
d[EQ]/dt &= k_{15}.[EPQ] + k_{18}.[Q].[E] - (k_{16}.[P] + k_{17}).[EQ]
\end{aligned} \tag{7.21}$$

The total concentration of enzyme is

$$[E_o] = [E] + [EB] + [EA] + [EP] + [EQ] + [EAB] + [EPQ] \tag{7.22}$$

The rate of production of Q which is called the rate of the reaction is

$$v = k_{11}.[EPQ] + k_{17}.[EQ] - (k_{18}.[E] + k_{12}.[EP]).Q \tag{7.23}$$

The solution of the above 8 equations was done using computer algebra technique, Gröbner bases and all the intermediates, EA, EB, EAB, EPQ, EP and EQ were eliminated to yield a formulae for v.

$$\begin{aligned}
v = \; & (E_o.([A]^2.[B].k_{11}.k_{13}.k_{16}.k_3.k_5.k_7.k_9.[P] + [A]^2.[B].k_{11}.k_{13}.k_{17}.k_3.k_5.k_7.k_9 \\
& +[A]^2.[B].k_{12}.k_{15}.k_{17}.k_3.k_5.k_7.k_9.[Q] + [A]^2.[B].k_{13}.k_{15}.k_{17}.k_3.k_5.k_7.k_9 \\
& +[A].[B]^2.k_1.k_{11}.k_{13}.k_{16}.k_3.k_7.k_9.[P] + [A].[B]^2.k_1.k_{11}.k_{13}.k_{17}.k_3.k_7.k_9 \\
& +[A].[B]^2.k_1.k_{12}.k_{15}.k_{17}.k_3.k_7.k_9.[Q] + [A].[B]^2.k_1.k_{13}.k_{15}.k_{17}.k_3.k_7.k_9 \\
& +[A].[B].k_1.k_{11}.k_{13}.k_{16}.k_3.k_6.k_9.[P] + [A].[B].k_1.k_{11}.k_{13}.k_{17}.k_3.k_6.k_9 \\
& +[A].[B].k_1.k_{12}.k_{15}.k_{17}.k_3.k_6.k_9.[Q] + [A].[B].k_1.k_{13}.k_{15}.k_{17}.k_3.k_6.k_9 \\
& +[A].[B].k_{11}.k_{13}.k_{16}.k_2.k_5.k_7.k_9.[P] + [A].[B].k_{11}.k_{13}.k_{17}.k_2.k_5.k_7.k_9 \\
& +[A].[B].k_{12}.k_{15}.k_{17}.k_2.k_5.k_7.k_9.[Q] + [A].[B].k_{13}.k_{15}.k_{17}.k_2.k_5.k_7.k_9 \\
& -[A].k_{10}.k_{12}.k_{14}.k_{16}.k_3.k_6.k_8.[P]^2.[Q] - [A].k_{10}.k_{12}.k_{14}.k_{17}.k_3.k_6.k_8.[P].[Q] \\
& -[A].k_{10}.k_{12}.k_{16}.k_{18}.k_3.k_6.k_8.[P].[Q]^2 - [A].k_{10}.k_{13}.k_{16}.k_{18}.k_3.k_6.k_8.[P].[Q] \\
& -[B].k_{10}.k_{12}.k_{14}.k_{16}.k_2.k_4.k_7.[P]^2.[Q] - [B].k_{10}.k_{12}.k_{14}.k_{17}.k_2.k_4.k_7.[P].[Q] \\
& -[B].k_{10}.k_{12}.k_{16}.k_{18}.k_2.k_4.k_7.[P].[Q]^2 - [B].k_{10}.k_{13}.k_{16}.k_{18}.k_2.k_4.k_7.[P].[Q] \\
& -k_{10}.k_{12}.k_{14}.k_{16}.k_2.k_4.k_6.[P]^2.[Q] - k_{10}.k_{12}.k_{14}.k_{16}.k_2.k_6.k_8.[P]^2.[Q] \\
& -k_{10}.k_{12}.k_{14}.k_{17}.k_2.k_4.k_6.[P].[Q] - k_{10}.k_{12}.k_{14}.k_{17}.k_2.k_6.k_8.[P].[Q] \\
& -k_{10}.k_{12}.k_{16}.k_{18}.k_2.k_4.k_6.[P].[Q]^2 - k_{10}.k_{12}.k_{16}.k_{18}.k_2.k_6.k_8.[P].[Q]^2 \\
& -k_{10}.k_{13}.k_{16}.k_{18}.k_2.k_4.k_6.[P].[Q] - k_{10}.k_{13}.k_{16}.k_{18}.k_2.k_6.k_8.[P].[Q])) \\
& /([A]^2.[B].k_{10}.k_{12}.k_{16}.k_3.k_5.k_7.[P].[Q] + [A]^2.[B].k_{10}.k_{12}.k_{17}.k_3.k_5.k_7.[Q] \\
& +[A]^2.[B].k_{10}.k_{13}.k_{16}.k_3.k_5.k_7.[P] + [A]^2.[B].k_{10}.k_{13}.k_{17}.k_3.k_5.k_7 \\
& +[A]^2.[B].k_{11}.k_{13}.k_{16}.k_3.k_5.k_7.[P] + [A]^2.[B].k_{11}.k_{13}.k_{17}.k_3.k_5.k_7 \\
& +[A]^2.[B].k_{11}.k_{16}.k_3.k_5.k_7.k_9.[P] + [A]^2.[B].k_{11}.k_{17}.k_3.k_5.k_7.k_9 \\
& +[A]^2.[B].k_{12}.k_{15}.k_{17}.k_3.k_5.k_7.[Q] + [A]^2.[B].k_{12}.k_{15}.k_3.k_5.k_7.k_9.[Q]
\end{aligned}$$

$$+[A]^2.[B].k_{12}.k_{16}.k_3.k_5.k_7.k_9.[P].[Q] + [A]^2.[B].k_{12}.k_{17}.k_3.k_5.k_7.k_9.[Q]$$
$$+[A]^2.[B].k_{13}.k_{15}.k_{17}.k_3.k_5.k_7 + [A]^2.[B].k_{13}.k_{15}.k_3.k_5.k_7.k_9$$
$$+[A]^2.[B].k_{13}.k_{16}.k_3.k_5.k_7.k_9.[P] + [A]^2.[B].k_{13}.k_{17}.k_3.k_5.k_7.k_9$$
$$+[A]^2.k_{10}.k_{12}.k_{16}.k_3.k_5.k_8.[P].[Q] + [A]^2.k_{10}.k_{12}.k_{17}.k_3.k_5.k_8.[Q]$$
$$+[A]^2.k_{10}.k_{13}.k_{16}.k_3.k_5.k_8.[P] + [A]^2.k_{10}.k_{13}.k_{17}.k_3.k_5.k_8$$
$$+[A]^2.k_{11}.k_{13}.k_{16}.k_3.k_5.k_8.[P] + [A]^2.k_{11}.k_{13}.k_{16}.k_3.k_5.k_9.[P]$$
$$+[A]^2.k_{11}.k_{13}.k_{17}.k_3.k_5.k_8 + [A]^2.k_{11}.k_{13}.k_{17}.k_3.k_5.k_9$$
$$+[A]^2.k_{12}.k_{15}.k_{17}.k_3.k_5.k_8.[Q] + [A]^2.k_{12}.k_{15}.k_{17}.k_3.k_5.k_9.[Q]$$
$$+[A]^2.k_{13}.k_{15}.k_{17}.k_3.k_5.k_8 + [A]^2.k_{13}.k_{15}.k_{17}.k_3.k_5.k_9$$
$$+[A].[B]^2.k_1.k_{10}.k_{12}.k_{16}.k_3.k_7.[P].[Q] + [A].[B]^2.k_1.k_{10}.k_{12}.k_{17}.k_3.k_7.[Q]$$
$$+[A].[B]^2.k_1.k_{10}.k_{13}.k_{16}.k_3.k_7.[P] + [A].[B]^2.k_1.k_{10}.k_{13}.k_{17}.k_3.k_7$$
$$+[A].[B]^2.k_1.k_{11}.k_{13}.k_{16}.k_3.k_7.[P] + [A].[B]^2.k_1.k_{11}.k_{13}.k_{17}.k_3.k_7$$
$$+[A].[B]^2.k_1.k_{11}.k_{16}.k_3.k_7.k_9.[P] + [A].[B]^2.k_1.k_{11}.k_{17}.k_3.k_7.k_9$$
$$+[A].[B]^2.k_1.k_{12}.k_{15}.k_{17}.k_3.k_7.[Q] + [A].[B]^2.k_1.k_{12}.k_{15}.k_3.k_7.k_9.[Q]$$
$$+[A].[B]^2.k_1.k_{12}.k_{16}.k_3.k_7.k_9.[P].[Q] + [A].[B]^2.k_1.k_{12}.k_{17}.k_3.k_7.k_9.[Q]$$
$$+[A].[B]^2.k_1.k_{13}.k_{15}.k_{17}.k_3.k_7 + [A].[B]^2.k_1.k_{13}.k_{15}.k_3.k_7.k_9$$
$$+[A].[B]^2.k_1.k_{13}.k_{16}.k_3.k_7.k_9.[P] + [A].[B]^2.k_1.k_{13}.k_{17}.k_3.k_7.k_9$$
$$+[A].[B].k_1.k_{10}.k_{12}.k_{16}.k_3.k_6.[P].[Q] + [A].[B].k_1.k_{10}.k_{12}.k_{16}.k_3.k_8.[P].[Q]$$
$$+[A].[B].k_1.k_{10}.k_{12}.k_{17}.k_3.k_6.[Q] + [A].[B].k_1.k_{10}.k_{12}.k_{17}.k_3.k_8.[Q]$$
$$+[A].[B].k_1.k_{10}.k_{13}.k_{16}.k_3.k_6.[P] + [A].[B].k_1.k_{10}.k_{13}.k_{16}.k_3.k_8.[P]$$
$$+[A].[B].k_1.k_{10}.k_{13}.k_{17}.k_3.k_6 + [A].[B].k_1.k_{10}.k_{13}.k_{17}.k_3.k_8$$
$$+[A].[B].k_1.k_{11}.k_{13}.k_{16}.k_3.k_6.[P] + [A].[B].k_1.k_{11}.k_{13}.k_{16}.k_3.k_8.[P]$$
$$+[A].[B].k_1.k_{11}.k_{13}.k_{17}.k_3.k_6 + [A].[B].k_1.k_{11}.k_{13}.k_{17}.k_3.k_8$$
$$+[A].[B].k_1.k_{11}.k_{16}.k_3.k_6.k_9.[P] + [A].[B].k_1.k_{11}.k_{17}.k_3.k_6.k_9$$
$$+[A].[B].k_1.k_{12}.k_{15}.k_{17}.k_3.k_6.[Q] + [A].[B].k_1.k_{12}.k_{15}.k_{17}.k_3.k_8.[Q]$$
$$+[A].[B].k_1.k_{12}.k_{15}.k_3.k_6.k_9.[Q] + [A].[B].k_1.k_{12}.k_{16}.k_3.k_6.k_9.[P].[Q]$$
$$+[A].[B].k_1.k_{12}.k_{17}.k_3.k_6.k_9.[Q] + [A].[B].k_1.k_{13}.k_{15}.k_{17}.k_3.k_6$$
$$+[A].[B].k_1.k_{13}.k_{15}.k_{17}.k_3.k_8 + [A].[B].k_1.k_{13}.k_{15}.k_3.k_6.k_9$$
$$+[A].[B].k_1.k_{13}.k_{16}.k_3.k_6.k_9.[P] + [A].[B].k_1.k_{13}.k_{17}.k_3.k_6.k_9$$

$$
\begin{aligned}
&+[A].[B].k_{10}.k_{12}.k_{14}.k_{16}.k_3.k_7.[P]^2.[Q] + [A].[B].k_{10}.k_{12}.k_{14}.k_{17}.k_3.k_7.[P].[Q] \\
&+[A].[B].k_{10}.k_{12}.k_{16}.k_{18}.k_3.k_7.[P].[Q]^2 + [A].[B].k_{10}.k_{12}.k_{16}.k_2.k_5.k_7.[P].[Q] \\
&+[A].[B].k_{10}.k_{12}.k_{16}.k_4.k_5.k_7.[P].[Q] + [A].[B].k_{10}.k_{12}.k_{17}.k_2.k_5.k_7.[Q] \\
&+[A].[B].k_{10}.k_{12}.k_{17}.k_4.k_5.k_7.[Q] + [A].[B].k_{10}.k_{13}.k_{16}.k_{18}.k_3.k_7.[P].[Q] \\
&+[A].[B].k_{10}.k_{13}.k_{16}.k_2.k_5.k_7.[P] + [A].[B].k_{10}.k_{13}.k_{16}.k_4.k_5.k_7.[P] \\
&+[A].[B].k_{10}.k_{13}.k_{17}.k_2.k_5.k_7 + [A].[B].k_{10}.k_{13}.k_{17}.k_4.k_5.k_7 \\
&+[A].[B].k_{11}.k_{13}.k_{16}.k_2.k_5.k_7.[P] + [A].[B].k_{11}.k_{13}.k_{16}.k_3.k_7.k_9.[P] \\
&+[A].[B].k_{11}.k_{13}.k_{16}.k_4.k_5.k_7.[P] + [A].[B].k_{11}.k_{13}.k_{17}.k_2.k_5.k_7 \\
&+[A].[B].k_{11}.k_{13}.k_{18}.k_3.k_7.k_9.[Q] + [A].[B].k_{11}.k_{14}.k_{16}.k_3.k_7.k_9.[P]^2 \\
&+[A].[B].k_{11}.k_{14}.k_{17}.k_3.k_7.k_9.[P] + [A].[B].k_{11}.k_{16}.k_{18}.k_3.k_7.k_9.[P].[Q] \\
&+[A].[B].k_{11}.k_{16}.k_2.k_5.k_7.k_9.[P] + [A].[B].k_{11}.k_{17}.k_2.k_5.k_7.k_9 \\
&+[A].[B].k_{12}.k_{14}.k_{15}.k_3.k_7.k_9.[P].[Q] + [A].[B].k_{12}.k_{14}.k_{16}.k_3.k_7.k_9.[P]^2.[Q] \\
&+[A].[B].k_{12}.k_{14}.k_{17}.k_3.k_7.k_9.[P].[Q] + [A].[B].k_{12}.k_{15}.k_{17}.k_2.k_5.k_7.[Q] \\
&+[A].[B].k_{12}.k_{15}.k_{17}.k_3.k_7.k_9.[Q] + [A].[B].k_{12}.k_{15}.k_{17}.k_4.k_5.k_7.[Q] \\
&+[A].[B].k_{12}.k_{15}.k_{18}.k_3.k_7.k_9.[Q]^2 + [A].[B].k_{12}.k_{15}.k_2.k_5.k_7.k_9.[Q] \qquad (7.24) \\
&+[A].[B].k_{12}.k_{16}.k_{18}.k_3.k_7.k_9.[P].[Q]^2 + [A].[B].k_{12}.k_{16}.k_2.k_5.k_7.k_9.[P].[Q] \\
&+[A].[B].k_{12}.k_{17}.k_2.k_5.k_7.k_9.[Q] + [A].[B].k_{13}.k_{15}.k_{17}.k_2.k_5.k_7 \\
&+[A].[B].k_{13}.k_{15}.k_{17}.k_3.k_7.k_9 + [A].[B].k_{13}.k_{15}.k_{17}.k_4.k_5.k_7 \\
&+[A].[B].k_{13}.k_{15}.k_{18}.k_3.k_7.k_9.[Q] + [A].[B].k_{13}.k_{15}.k_2.k_5.k_7.k_9 \\
&+[A].[B].k_{13}.k_{16}.k_{18}.k_3.k_7.k_9.[P].[Q] + [A].[B].k_{13}.k_{16}.k_2.k_5.k_7.k_9.[P] \\
&+[A].[B].k_{13}.k_{17}.k_2.k_5.k_7.k_9 + [A].[B].k_{14}.k_{15}.k_{17}.k_3.k_7.k_9.[P] \\
&+[A].k_{10}.k_{12}.k_{14}.k_{16}.k_3.k_6.[P]^2.[Q] + [A].k_{10}.k_{12}.k_{14}.k_{16}.k_3.k_8.[P]^2.[Q] \\
&+[A].k_{10}.k_{12}.k_{14}.k_{17}.k_3.k_6.[P].[Q] + [A].k_{10}.k_{12}.k_{14}.k_{17}.k_3.k_8.[P].[Q] \\
&+[A].k_{10}.k_{12}.k_{16}.k_{18}.k_3.k_6.[P].[Q]^2 + [A].k_{10}.k_{12}.k_{16}.k_{18}.k_3.k_8.[P].[Q]^2 \\
&+[A].k_{10}.k_{12}.k_{16}.k_2.k_4.k_5.[P].[Q] + [A].k_{10}.k_{12}.k_{16}.k_2.k_5.k_8.[P].[Q] \\
&+[A].k_{10}.k_{12}.k_{16}.k_3.k_6.k_8.[P].[Q] + [A].k_{10}.k_{12}.k_{17}.k_2.k_4.k_5.[Q] \\
&+[A].k_{10}.k_{12}.k_{17}.k_2.k_5.k_8.[Q] + [A].k_{10}.k_{12}.k_{17}.k_3.k_6.k_8.[Q] \\
&+[A].k_{10}.k_{12}.k_{18}.k_3.k_6.k_8.[Q]^2 + [A].k_{10}.k_{13}.k_{16}.k_{18}.k_3.k_6.[P].[Q] \\
&+[A].k_{10}.k_{13}.k_{16}.k_{18}.k_3.k_8.[P].[Q] + [A].k_{10}.k_{13}.k_{16}.k_2.k_4.k_5.[P]
\end{aligned}
$$

$$+[A].k_{10}.k_{13}.k_{16}.k_2.k_5.k_8.[P] + [A].k_{10}.k_{13}.k_{16}.k_3.k_6.k_8.[P]$$
$$+[A].k_{10}.k_{13}.k_{17}.k_2.k_4.k_5 + [A].k_{10}.k_{13}.k_{17}.k_2.k_5.k_8$$
$$+[A].k_{10}.k_{13}.k_{17}.k_3.k_6.k_8 + [A].k_{10}.k_{13}.k_{18}.k_3.k_6.k_8.[Q]$$
$$+[A].k_{10}.k_{14}.k_{16}.k_3.k_6.k_8.[P]^2 + [A].k_{10}.k_{14}.k_{17}.k_3.k_6.k_8.[P]$$
$$+[A].k_{11}.k_{13}.k_{16}.k_2.k_4.k_5.[P] + [A].k_{11}.k_{13}.k_{16}.k_2.k_5.k_8.[P]$$
$$+[A].k_{11}.k_{13}.k_{16}.k_2.k_5.k_9.[P] + [A].k_{11}.k_{13}.k_{16}.k_3.k_6.k_8.[P]$$
$$+[A].k_{11}.k_{13}.k_{16}.k_3.k_6.k_9.[P] + [A].k_{11}.k_{13}.k_{17}.k_2.k_4.k_5$$
$$+[A].k_{11}.k_{13}.k_{17}.k_2.k_5.k_8 + [A].k_{11}.k_{13}.k_{17}.k_2.k_5.k_9$$
$$+[A].k_{11}.k_{13}.k_{17}.k_3.k_6.k_8 + [A].k_{11}.k_{13}.k_{17}.k_3.k_6.k_9$$
$$+[A].k_{11}.k_{13}.k_{18}.k_3.k_6.k_8.[Q] + [A].k_{11}.k_{13}.k_{18}.k_3.k_6.k_9.[Q]$$
$$+[A].k_{11}.k_{14}.k_{16}.k_3.k_6.k_8.[P]^2 + [A].k_{11}.k_{14}.k_{16}.k_3.k_6.k_9.[P]^2$$
$$+[A].k_{11}.k_{14}.k_{17}.k_3.k_6.k_8.[P] + [A].k_{11}.k_{14}.k_{17}.k_3.k_6.k_9.[P]$$
$$+[A].k_{11}.k_{16}.k_{18}.k_3.k_6.k_8.[P].[Q] + [A].k_{11}.k_{16}.k_{18}.k_3.k_6.k_9.[P].[Q]$$
$$+[A].k_{12}.k_{14}.k_{15}.k_3.k_6.k_8.[P].[Q] + [A].k_{12}.k_{14}.k_{15}.k_3.k_6.k_9.[P].[Q]$$
$$+[A].k_{12}.k_{14}.k_{16}.k_3.k_6.k_8.[P]^2.[Q] + [A].k_{12}.k_{14}.k_{16}.k_3.k_6.k_9.[P]^2.[Q]$$
$$+[A].k_{12}.k_{14}.k_{17}.k_3.k_6.k_8.[P].[Q] + [A].k_{12}.k_{14}.k_{17}.k_3.k_6.k_9.[P].[Q]$$
$$+[A].k_{12}.k_{15}.k_{17}.k_2.k_4.k_5.[Q] + [A].k_{12}.k_{15}.k_{17}.k_2.k_5.k_8.[Q]$$
$$+[A].k_{12}.k_{15}.k_{17}.k_2.k_5.k_9.[Q] + [A].k_{12}.k_{15}.k_{17}.k_3.k_6.k_8.[Q]$$
$$+[A].k_{12}.k_{15}.k_{17}.k_3.k_6.k_9.[Q] + [A].k_{12}.k_{15}.k_{18}.k_3.k_6.k_8.[Q]^2$$
$$+[A].k_{12}.k_{15}.k_{18}.k_3.k_6.k_9.[Q]^2 + [A].k_{12}.k_{16}.k_{18}.k_3.k_6.k_8.[P].[Q]^2$$
$$+[A].k_{12}.k_{16}.k_{18}.k_3.k_6.k_9.[P].[Q]^2 + [A].k_{13}.k_{15}.k_{17}.k_2.k_4.k_5$$
$$+[A].k_{13}.k_{15}.k_{17}.k_2.k_5.k_8 + [A].k_{13}.k_{15}.k_{17}.k_2.k_5.k_9$$
$$+[A].k_{13}.k_{15}.k_{17}.k_3.k_6.k_8 + [A].k_{13}.k_{15}.k_{17}.k_3.k_6.k_9$$
$$+[A].k_{13}.k_{15}.k_{18}.k_3.k_6.k_8.[Q] + [A].k_{13}.k_{15}.k_{18}.k_3.k_6.k_9.[Q]$$
$$+[A].k_{13}.k_{16}.k_{18}.k_3.k_6.k_8.[P].[Q] + [A].k_{13}.k_{16}.k_{18}.k_3.k_6.k_9.[P].[Q]$$
$$+[A].k_{14}.k_{15}.k_{17}.k_3.k_6.k_8.[P] + [A].k_{14}.k_{15}.k_{17}.k_3.k_6.k_9.[P]$$
$$+[B]^2.k_1.k_{10}.k_{12}.k_{16}.k_4.k_7.[P].[Q] + [B]^2.k_1.k_{10}.k_{12}.k_{17}.k_4.k_7.[Q]$$
$$+[B]^2.k_1.k_{10}.k_{13}.k_{16}.k_4.k_7.[P] + [B]^2.k_1.k_{10}.k_{13}.k_{17}.k_4.k_7$$
$$+[B]^2.k_1.k_{11}.k_{13}.k_{16}.k_4.k_7.[P] + [B]^2.k_1.k_{11}.k_{13}.k_{16}.k_7.k_9.[P]$$

$$+[B]^2.k_1.k_{11}.k_{13}.k_{17}.k_4.k_7 + [B]^2.k_1.k_{11}.k_{13}.k_{17}.k_7.k_9$$
$$+[B]^2.k_1.k_{12}.k_{15}.k_{17}.k_4.k_7.[Q] + [B]^2.k_1.k_{12}.k_{15}.k_{17}.k_7.k_9.[Q]$$
$$+[B]^2.k_1.k_{13}.k_{15}.k_{17}.k_4.k_7 + [B]^2.k_1.k_{13}.k_{15}.k_{17}.k_7.k_9$$
$$+[B].k_1.k_{10}.k_{12}.k_{16}.k_4.k_6.[P].[Q] + [B].k_1.k_{10}.k_{12}.k_{16}.k_6.k_8.[P].[Q]$$
$$+[B].k_1.k_{10}.k_{12}.k_{17}.k_4.k_6.[Q] + [B].k_1.k_{10}.k_{12}.k_{17}.k_6.k_8.[Q]$$
$$+[B].k_1.k_{10}.k_{13}.k_{16}.k_4.k_6.[P] + [B].k_1.k_{10}.k_{13}.k_{16}.k_6.k_8.[P]$$
$$+[B].k_1.k_{10}.k_{13}.k_{17}.k_4.k_6 + [B].k_1.k_{10}.k_{13}.k_{17}.k_6.k_8$$
$$+[B].k_1.k_{11}.k_{13}.k_{16}.k_4.k_6.[P] + [B].k_1.k_{11}.k_{13}.k_{16}.k_6.k_8.[P]$$
$$+[B].k_1.k_{11}.k_{13}.k_{16}.k_6.k_9.[P] + [B].k_1.k_{11}.k_{13}.k_{17}.k_4.k_6$$
$$+[B].k_1.k_{11}.k_{13}.k_{17}.k_6.k_8 + [B].k_1.k_{11}.k_{13}.k_{17}.k_6.k_9$$
$$+[B].k_1.k_{12}.k_{15}.k_{17}.k_4.k_6.[Q] + [B].k_1.k_{12}.k_{15}.k_{17}.k_6.k_8.[Q]$$
$$+[B].k_1.k_{12}.k_{15}.k_{17}.k_6.k_9.[Q] + [B].k_1.k_{13}.k_{15}.k_{17}.k_4.k_6$$
$$+[B].k_1.k_{13}.k_{15}.k_{17}.k_6.k_8 + [B].k_1.k_{13}.k_{15}.k_{17}.k_6.k_9$$
$$+[B].k_{10}.k_{12}.k_{14}.k_{16}.k_2.k_7.[P]^2.[Q] + [B].k_{10}.k_{12}.k_{14}.k_{16}.k_4.k_7.[P]^2.[Q]$$
$$+[B].k_{10}.k_{12}.k_{14}.k_{17}.k_2.k_7.[P].[Q] + [B].k_{10}.k_{12}.k_{14}.k_{17}.k_4.k_7.[P].[Q]$$
$$+[B].k_{10}.k_{12}.k_{16}.k_{18}.k_2.k_7.[P].[Q]^2 + [B].k_{10}.k_{12}.k_{16}.k_{18}.k_4.k_7.[P].[Q]^2$$
$$+[B].k_{10}.k_{12}.k_{16}.k_2.k_4.k_7.[P].[Q] + [B].k_{10}.k_{12}.k_{17}.k_2.k_4.k_7.[Q]$$
$$+[B].k_{10}.k_{12}.k_{18}.k_2.k_4.k_7.[Q]^2 + [B].k_{10}.k_{13}.k_{16}.k_{18}.k_2.k_7.[P].[Q]$$
$$+[B].k_{10}.k_{13}.k_{16}.k_{18}.k_4.k_7.[P].[Q] + [B].k_{10}.k_{13}.k_{16}.k_2.k_4.k_7.[P]$$
$$+[B].k_{10}.k_{13}.k_{17}.k_2.k_4.k_7 + [B].k_{10}.k_{13}.k_{18}.k_2.k_4.k_7.[Q]$$
$$+[B].k_{10}.k_{14}.k_{16}.k_2.k_4.k_7.[P]^2 + [B].k_{10}.k_{14}.k_{17}.k_2.k_4.k_7.[P]$$
$$+[B].k_{11}.k_{13}.k_{16}.k_2.k_4.k_7.[P] + [B].k_{11}.k_{13}.k_{16}.k_2.k_7.k_9.[P]$$
$$+[B].k_{11}.k_{13}.k_{17}.k_2.k_4.k_7 + [B].k_{11}.k_{13}.k_{17}.k_2.k_7.k_9$$
$$+[B].k_{11}.k_{13}.k_{18}.k_2.k_4.k_7.[Q] + [B].k_{11}.k_{13}.k_{18}.k_2.k_7.k_9.[Q]$$
$$+[B].k_{11}.k_{14}.k_{16}.k_2.k_4.k_7.[P]^2 + [B].k_{11}.k_{14}.k_{16}.k_2.k_7.k_9.[P]^2$$
$$+[B].k_{11}.k_{14}.k_{17}.k_2.k_4.k_7.[P] + [B].k_{11}.k_{14}.k_{17}.k_2.k_7.k_9.[P]$$
$$+[B].k_{11}.k_{16}.k_{18}.k_2.k_4.k_7.[P].[Q] + [B].k_{11}.k_{16}.k_{18}.k_2.k_7.k_9.[P].[Q]$$
$$+[B].k_{12}.k_{14}.k_{15}.k_2.k_4.k_7.[P].[Q] + [B].k_{12}.k_{14}.k_{15}.k_2.k_7.k_9.[P].[Q]$$
$$+[B].k_{12}.k_{14}.k_{16}.k_2.k_4.k_7.[P]^2.[Q] + [B].k_{12}.k_{14}.k_{16}.k_2.k_7.k_9.[P]^2.[Q]$$

$$+[B].k_{12}.k_{14}.k_{17}.k_2.k_4.k_7.[P].[Q] + [B].k_{12}.k_{14}.k_{17}.k_2.k_7.k_9.[P].[Q]$$
$$+[B].k_{12}.k_{15}.k_{17}.k_2.k_4.k_7.[Q] + [B].k_{12}.k_{15}.k_{17}.k_2.k_7.k_9.[Q]$$
$$+[B].k_{12}.k_{15}.k_{18}.k_2.k_4.k_7.[Q]^2 + [B].k_{12}.k_{15}.k_{18}.k_2.k_7.k_9.[Q]^2$$
$$+[B].k_{12}.k_{16}.k_{18}.k_2.k_4.k_7.[P].[Q]^2 + [B].k_{12}.k_{16}.k_{18}.k_2.k_7.k_9.[P].[Q]^2$$
$$+[B].k_{13}.k_{15}.k_{17}.k_2.k_4.k_7 + [B].k_{13}.k_{15}.k_{17}.k_2.k_7.k_9$$
$$+[B].k_{13}.k_{15}.k_{18}.k_2.k_4.k_7.[Q] + [B].k_{13}.k_{15}.k_{18}.k_2.k_7.k_9.[Q]$$
$$+[B].k_{13}.k_{16}.k_{18}.k_2.k_4.k_7.[P].[Q] + [B].k_{13}.k_{16}.k_{18}.k_2.k_7.k_9.[P].[Q]$$
$$+[B].k_{14}.k_{15}.k_{17}.k_2.k_4.k_7.[P] + [B].k_{14}.k_{15}.k_{17}.k_2.k_7.k_9.[P]$$
$$+k_{10}.k_{12}.k_{14}.k_{16}.k_2.k_6.[P]^2.[Q] + k_{10}.k_{12}.k_{14}.k_{16}.k_2.k_8.[P]^2.[Q]$$
$$+k_{10}.k_{12}.k_{14}.k_{16}.k_4.k_6.[P]^2.[Q] + k_{10}.k_{12}.k_{14}.k_{17}.k_2.k_6.[P].[Q]$$
$$+k_{10}.k_{12}.k_{14}.k_{17}.k_2.k_8.[P].[Q] + k_{10}.k_{12}.k_{14}.k_{17}.k_4.k_6.[P].[Q]$$
$$+k_{10}.k_{12}.k_{16}.k_{18}.k_2.k_6.[P].[Q]^2 + k_{10}.k_{12}.k_{16}.k_{18}.k_2.k_8.[P].[Q]^2$$
$$+k_{10}.k_{12}.k_{16}.k_{18}.k_4.k_6.[P].[Q]^2 + k_{10}.k_{12}.k_{16}.k_2.k_4.k_6.[P].[Q]$$
$$+k_{10}.k_{12}.k_{16}.k_2.k_6.k_8.[P].[Q] + k_{10}.k_{12}.k_{17}.k_2.k_4.k_6.[Q]$$
$$+k_{10}.k_{12}.k_{17}.k_2.k_6.k_8.[Q] + k_{10}.k_{12}.k_{18}.k_2.k_4.k_6.[Q]^2$$
$$+k_{10}.k_{12}.k_{18}.k_2.k_6.k_8.[Q]^2 + k_{10}.k_{13}.k_{16}.k_{18}.k_2.k_6.[P].[Q]$$
$$+k_{10}.k_{13}.k_{16}.k_{18}.k_2.k_8.[P].[Q] + k_{10}.k_{13}.k_{16}.k_{18}.k_4.k_6.[P].[Q]$$
$$+k_{10}.k_{13}.k_{16}.k_2.k_4.k_6.[P] + k_{10}.k_{13}.k_{16}.k_2.k_6.k_8.[P]$$
$$+k_{10}.k_{13}.k_{17}.k_2.k_4.k_6 + k_{10}.k_{13}.k_{17}.k_2.k_6.k_8$$
$$+k_{10}.k_{13}.k_{18}.k_2.k_4.k_6.[Q] + k_{10}.k_{13}.k_{18}.k_2.k_6.k_8.[Q]$$
$$+k_{10}.k_{14}.k_{16}.k_2.k_4.k_6.[P]^2 + k_{10}.k_{14}.k_{16}.k_2.k_6.k_8.[P]^2$$
$$+k_{10}.k_{14}.k_{17}.k_2.k_4.k_6.[P] + k_{10}.k_{14}.k_{17}.k_2.k_6.k_8.[P]$$
$$+k_{11}.k_{13}.k_{16}.k_2.k_4.k_6.[P] + k_{11}.k_{13}.k_{16}.k_2.k_6.k_8.[P]$$
$$+k_{11}.k_{13}.k_{16}.k_2.k_6.k_9.[P] + k_{11}.k_{13}.k_{17}.k_2.k_4.k_6$$
$$+k_{11}.k_{13}.k_{17}.k_2.k_6.k_8 + k_{11}.k_{13}.k_{17}.k_2.k_6.k_9$$
$$+k_{11}.k_{13}.k_{18}.k_2.k_4.k_6.[Q] + k_{11}.k_{13}.k_{18}.k_2.k_6.k_8.[Q]$$
$$+k_{11}.k_{13}.k_{18}.k_2.k_6.k_9.[Q] + k_{11}.k_{14}.k_{16}.k_2.k_4.k_6.[P]^2$$
$$+k_{11}.k_{14}.k_{16}.k_2.k_6.k_8.[P]^2 + k_{11}.k_{14}.k_{16}.k_2.k_6.k_9.[P]^2$$

$$+k_{11}.k_{14}.k_{17}.k_2.k_4.k_6.[P] + k_{11}.k_{14}.k_{17}.k_2.k_6.k_8.[P]$$
$$+k_{11}.k_{14}.k_{17}.k_2.k_6.k_9.[P] + k_{11}.k_{16}.k_{18}.k_2.k_4.k_6.[P].[Q]$$
$$+k_{11}.k_{16}.k_{18}.k_2.k_6.k_8.[P].[Q] + k_{11}.k_{16}.k_{18}.k_2.k_6.k_9.[P].[Q]$$
$$+k_{12}.k_{14}.k_{15}.k_2.k_4.k_6.[P].[Q] + k_{12}.k_{14}.k_{15}.k_2.k_6.k_8.[P].[Q]$$
$$+k_{12}.k_{14}.k_{15}.k_2.k_6.k_9.[P].[Q] + k_{12}.k_{14}.k_{16}.k_2.k_4.k_6.[P]^2.[Q]$$
$$+k_{12}.k_{14}.k_{16}.k_2.k_6.k_8.[P]^2.[Q] + k_{12}.k_{14}.k_{16}.k_2.k_6.k_9.[P]^2.[Q]$$
$$+k_{12}.k_{14}.k_{17}.k_2.k_4.k_6.[P].[Q] + k_{12}.k_{14}.k_{17}.k_2.k_6.k_8.[P].[Q]$$
$$+k_{12}.k_{14}.k_{17}.k_2.k_6.k_9.[P].[Q] + k_{12}.k_{15}.k_{17}.k_2.k_4.k_6.[Q]$$
$$+k_{12}.k_{15}.k_{17}.k_2.k_6.k_8.[Q] + k_{12}.k_{15}.k_{17}.k_2.k_6.k_9.[Q]$$
$$+k_{12}.k_{15}.k_{18}.k_2.k_4.k_6.[Q]^2 + k_{12}.k_{15}.k_{18}.k_2.k_6.k_8.[Q]^2$$
$$+k_{12}.k_{15}.k_{18}.k_2.k_6.k_9.[Q]^2 + k_{12}.k_{16}.k_{18}.k_2.k_4.k_6.[P].[Q]^2$$
$$+k_{12}.k_{16}.k_{18}.k_2.k_6.k_8.[P].[Q]^2 + k_{12}.k_{16}.k_{18}.k_2.k_6.k_9.[P].[Q]^2$$
$$+k_{13}.k_{15}.k_{17}.k_2.k_4.k_6 + k_{13}.k_{15}.k_{17}.k_2.k_6.k_8$$
$$+k_{13}.k_{15}.k_{17}.k_2.k_6.k_9 + k_{13}.k_{15}.k_{18}.k_2.k_4.k_6.[Q]$$
$$+k_{13}.k_{15}.k_{18}.k_2.k_6.k_8.[Q] + k_{13}.k_{15}.k_{18}.k_2.k_6.k_9.[Q]$$
$$+k_{13}.k_{16}.k_{18}.k_2.k_4.k_6.[P].[Q] + k_{13}.k_{16}.k_{18}.k_2.k_6.k_8.[P].[Q]$$
$$+k_{13}.k_{16}.k_{18}.k_2.k_6.k_9.[P].[Q] + k_{14}.k_{15}.k_{17}.k_2.k_4.k_6.[P]$$
$$+k_{14}.k_{15}.k_{17}.k_2.k_6.k_8.[P] + k_{14}.k_{15}.k_{17}.k_2.k_6.k_9.[P])$$

The Computer algebra system provides a fast and easy to use way of obtaining rate-laws of arbitrarily complex enzymatic mechanisms. This is a feature that could not be achieved if the King-Altman method was to be used as it does not handle several mechanisms [King and Altman, 1956]. These mechanisms need not be very complex. Indeed a good example is the simple Michaelis-Menten mechanism which only contains two different enzymatic forms. Moreover, in the case of very complex mechanisms has difficulty in finding all patterns that will generate the denominator of the rate equation for King-Altman. Our method is general purpose. The second example could not be done by King-Altman method.

7.2 Automatic Derivation of Conservation Relationships

The derivation of the concentration equations which required for the elimination of the intermediate variables in the given metabolic network is straightforward, but might not be so for more intricate models. Fortunately a general ideas have already been expressed by people working on metabolic network [Reder, 1988 and Letellier et al, 1991]. Nevertheless, we have computerised the Reder's algorithm to avoid logical errors.

A program which was given in appendix **B** takes a list of metabolites, **X**, and a stoichiometry matrix, **N**, and determinates conservation relationship of the given metabolic system.

The matrix **N** (see section 4.2) is partitioned into a generating basis matrix, $\mathbf{N_R}$, and a residual matrix, $\mathbf{N_o}$. The rows of $\mathbf{N_R}$ are linearly independent and the rows of $\mathbf{N_o}$ can all be generated as linear combinations of $\mathbf{N_R}$. Having obtained $\mathbf{N_R}$ and $\mathbf{N_o}$ we construct the matrix $\mathbf{L_o}$ which is used to calculate the conservation relationship.

Let the stoichiometry matrix **N** be the matrix of an application from an m-dimensional space to a r-dimensional space, i.e. **N** is of the form $\mathbf{N}_{m\times r}$ and the rank of the $\mathbf{N}_{m\times r}$ be m_o. The rank of **N**, m_o, is found using REDUCE's **rank** command. Finding conservation relationship is done using following algorithm:

1. Calculate the matrix $\mathbf{N_R}$ and $\mathbf{N_o}$. The matrix $\mathbf{N_R}$ is extracted from **N** by taking the m_o linear independent rows of **N**, and the matrix $\mathbf{N_o}$ is the residual matrix of **N**. Therefore, dimensions of matrix $\mathbf{N_R}$ and $\mathbf{N_o}$ are $m_o \times r$ and $(m - m_o) \times r$, respectively.

2. Partition the matrix $\mathbf{N_R}$ into $\mathbf{N_{R_o}}$ and $\mathbf{N_{R_1}}$. The matrix $\mathbf{N_{R_o}}$ is extracted from matrix $\mathbf{N_R}$ by taking m_o independent columns of $\mathbf{N_R}$. Therefore, $\mathbf{N_{R_o}}$ is an invertible $(m_o \times m_o)$ square matrix and $\mathbf{N_{R_1}}$ is a $(m_o \times (r - m_o))$ matrix.

3. Partition the matrix $\mathbf{N_o}$ into $\mathbf{N_o^*}$ and $\mathbf{N_1^*}$. The matrix $\mathbf{N_o^*}$ is built by taking same independent columns of $\mathbf{N_o}$, i.e. $\mathbf{N_o^*}$ is $((m - m_o) \times m_o)$ matrix, thus $\mathbf{N_1^*}$ is a $((m - m_o) \times (r - m_o))$ matrix.

4. Calculate the matrix **Lo** using matrixes $\mathbf{N_{R_o}}$ and $\mathbf{N_o^*}$ such as

$$\mathbf{L_o} = \mathbf{N_o^*} \times \mathbf{N_{R_o}^{-1}} \tag{7.25}$$

5. Decompose the concentration matrix **X** into its first m_o rows $\mathbf{X_R}$ and its $(m - m_o)$ last rows $\mathbf{X_R'}$.

6. Conservation relationship is for given metabolic network is:

$$\frac{\partial}{\partial t}(\mathbf{X_R'} - \mathbf{L_o} \times \mathbf{X_R}) = 0 \tag{7.26}$$

Our technique has been unlimited to handle more complex metabolic network such as branched and cycle. The advantage of the use of the technique is to avoid logical errors.

7.3 Automatic Derivation of Metabolic Control Coefficients

Metabolic control analysis allows one to quantify the behaviour of a metabolic pathway in steady state in terms of dimensionless coefficients. From the definition of metabolite and flux control coefficients and elasticities we are able to derive symbolic forms of these parameters, in terms of conventional kinetic parameters. At the simplest level we are able to substitute values of these kinetic parameters, to yield values for the metabolic control coefficients. Since we are substituting into symbolic equations we can always guarantee the conservation relationships hold. We have done this in chapter 6 for the two enzyme system in *vitro* using aspartate aminotransferase-malate dehydrogenase coupled system.

The basic relationships are the summation and connectivity theorems. They allow one to express the behaviour of the system variables in terms of the kinetic properties of the isolated enzymatic reactions that build up the metabolic network. A matrix method was derived [Fell and Sauro, 1985; Sauro *et al*, 1987] that allows the determination of

the flux and concentration control coefficients of enzymes from their kinetic properties represented by the elasticity coefficients [Reder, 1988 and Letellier *et al*, 1991; Kacser and Porteous, 1987; Kacser *et al*, 1990; Small *et al*, 1989; Delgado *et al*, 1992].

The ability to define the control coefficient equations in matrix form not only allows easy solution by numerical inversion but also opens up the possibility of obtaining the algebraic solutions by symbolic manipulation of the matrix. However for matrixes larger than rank 4 or 5, this latter possibility, if done by hand, becomes very tedious and is prone to error. The solution to this problem is to develop computer software to automatically carry out this procedure. In this section a brief description of such a program is given and based on Reder method [Letellier *et al*, 1991]. The program was written in algebraic and symbolic REDUCE form. Using this program it is possible to calculate the control coefficients from elasticity coefficients whatever the metabolic network.

We consider a metabolic system such as,

$$\underset{e_1}{\overset{v_1}{\rightleftharpoons}} X_1 \underset{e_2}{\overset{v_2}{\rightleftharpoons}} X_2 \underset{e_3}{\overset{v_3}{\rightleftharpoons}} \cdots \underset{e_{r-1}}{\overset{v_{r-1}}{\rightleftharpoons}} X_m \underset{e_r}{\overset{v_r}{\rightleftharpoons}}$$

that is, a sequence of intermediary metabolites $(x_1, x_2, \cdots, x_m)$ and a squence of catalysts (e_i). The system is described by a system of differential equations

$$\frac{dx}{dt} = \mathbf{N} \times v \tag{7.27}$$

where x is column vector of concentrations of metabolites x_i, t the independent variable time, v the column vector of the rates v_i, and $\mathbf{N_{mxr}}$ the stoichiometry matrix. The concentrations of the metabolites $(x_1, x_2, \cdots, x_m)$ at steady state are a solution of

$$\mathbf{N}\mathbf{v}(x_1^o, \cdots, x_m^o, e_i) = 0 \tag{7.28}$$

and the fluxes at this steady state are defined by:

$$J_i = v_i(x_1^o, \cdots, x_m^o, e_i). \tag{7.29}$$

The brief description of the Reder's method as follows;

The stoichiometric matrix **N** can be decomposed as;

$$\mathbf{N} = \mathbf{L} \times N_R \tag{7.30}$$

As we defined $\mathbf{N_R}$ in previous section it is an $m_o \times r$ matrix formed by the first m_o rows of **N** that constitute a basis for its row space. **L** is an $m \times m_o$ matrix that has the form

$$\mathbf{L} = \begin{bmatrix} \mathbf{I_{m_o}} \\ \mathbf{L_o} \end{bmatrix} \tag{7.31}$$

where $\mathbf{I_{m_o}}$ is the $m_o \times m_o$ identity matrix and **Lo** is $(m - m_o) \times m_o$.

The elasticity matrix E' is defined by:

$$\mathbf{E'} = \begin{bmatrix} \epsilon'_{11} & \cdots & \epsilon'_{1m} \\ \cdots & \cdots & \cdots\cdots \\ \epsilon'_{r1} & \cdots & \epsilon'_{rm} \end{bmatrix} \tag{7.32}$$

where ϵ'_{ij} is

$$\epsilon'_{ij} = \frac{\delta v_i}{\delta x_j} \tag{7.33}$$

The matrix of flux control coefficients $\mathbf{C'}$ is calculated using:

$$\mathbf{C'} = \mathbf{I} - \mathbf{E'.L.[N_R.E'.L]^{-1}.N_R} \tag{7.34}$$

where **I** is the identity matrix with dimension $r \times r$. The elements of the j-*th* column of the $\mathbf{C'}$ matrix is in the usual form:

$$\mathbf{C'_{kj}} = \frac{\delta J_k / \delta e_j}{\delta v_j / \delta e_j} \tag{7.35}$$

The metabolite control coefficient matrix $\mathbf{S'}$ is calculated using following relationship:

$$\mathbf{S'} = -\mathbf{L.[N_R.E'.L]^{-1}.N_R} \tag{7.36}$$

The elements of the J-*th* column of the $\mathbf{S}'$ matrix is written in the usual form:

$$\mathbf{S'_{kj}} = \frac{\delta x_k / \delta e_i}{\delta v_j / \delta e_i} \tag{7.37}$$

The summation relationships between the flux control coefficients are derived from the symbolic development of the equation:

$$\mathbf{C'.K} = \mathbf{K} \tag{7.38}$$

where $\mathbf{K}$ is a matrix which contains the vectors of a basis of the null-space of the $\mathbf{N}$ matrix. We will show how to obtain the matrix $\mathbf{K}$ in this section later.

The summation relationships between the metabolite control coefficients;

$$\mathbf{S'.K} = \mathbf{0} \tag{7.39}$$

The connectivity relationships between the flux control coefficients and the elasticity coefficients are

$$\mathbf{C'.(E'.L)} = \mathbf{0} \tag{7.40}$$

Similarly the connectivity relationships between the metabolite control coefficients and the elasticity coefficients are

$$\mathbf{S'.(E'.L)} = -\mathbf{L} \tag{7.41}$$

We used here nonnormalised coefficients, however it is easy to transform them into normalised coefficients using the formulae:

$$\begin{aligned} \mathbf{C_{ij}} &= \mathbf{C'_{ij}}.v_j/v_i \\ \mathbf{S_{ij}} &= \mathbf{S'_{ij}}.v_j/x_i \\ \epsilon_{ij} &= \epsilon'_{ij}.x_j/v_i \end{aligned} \tag{7.42}$$

The matrix $\mathbf{L}$ is determined with m_o independent columns of $\mathbf{N}$, i.e. on matrix $\mathbf{N^*}$ and $\mathbf{N_{R_o}}$ such as

$$\mathbf{L} = \mathbf{N^*} \times \mathbf{N_{R_o}^{-1}} \tag{7.43}$$

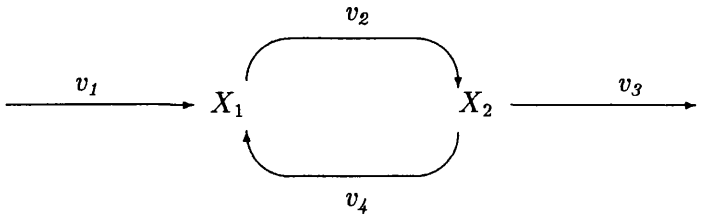


Figure 7-1: A substrate cycle model in a simple pathway.

The matrix **K** is a matrix whose columns form a basis for the kernel of **N**, i.e. the columns of the **K** matrix form a basis of the null-space of **N**. Which is also the null-space of $\mathbf{N_R}$. The **K** matrix is an $(r \times (r - m_o))$ matrix. Its columns are built to be independent by looking for a **K** matrix of the form

$$\mathbf{K} = \begin{bmatrix} \mathbf{A} \\ \mathbf{I_{r-m_o}} \end{bmatrix} \tag{7.44}$$

Note that $\mathbf{N.K} = \mathbf{N_R.K} = \mathbf{0}$ by definition of **K** [Letellier et al, 1991]. If **A** exists, it then verifies

$$\mathbf{N_{R_o}.A} + \mathbf{N_{R_1}} = 0 \tag{7.45}$$

Thus,

$$\mathbf{A} = \mathbf{N_{R_o}^{-1}}.(-\mathbf{N_R}). \tag{7.46}$$

Our program always uses modified Gram-Schmidt orthogonalisation process [Simmons, 1963] to find independent rows or columns of the matrixes such as $\mathbf{N_R}$ and $\mathbf{N_{R_o}}$ we needed, and computes the nonnormalised coefficients $\mathbf{C_{ij}}$, $\mathbf{S_{ij}}$ and ϵ_{ij}.

7.3.1 An Example of Application of the Computer Program

Let us consider a substrate cycle model, figure 7-1, the rate law for each enzyme

is a fully reversible Michaelis-Menten mechanism. The matrix N for the mechanism is:

$$N = \begin{bmatrix} 1 & -1 & 0 & 1 \\ 0 & 1 & -1 & -1 \end{bmatrix} \tag{7.47}$$

The computer program calculates flux control coefficients, metabolite control coefficients, summation and connectivity relationships in terms of elasticity coefficients from matrix N for given metabolite network. The above mechanism computer result as follows;

```
{{ Matrix N  was changed and New N is ,
                    [1  -1  1   0 ]
                    [0  1   -1  -1]
     Flux Control Coefficients in matrix form,
MAT(((ELS(2,4)*(ELS(1,3) - ELS(1,2)))/(ELS(2,4)*ELS(1,3)
      - ELS(2,4)*ELS(1,2) + ELS(2,4)*ELS(1,1)
      + ELS(2,3)*ELS(1,1) - ELS(2,2)*ELS(1,1)),(ELS(2,4)
     *ELS(1,1))/(ELS(2,4)*ELS(1,3) - ELS(2,4)*ELS(1,2)
      + ELS(2,4)*ELS(1,1) + ELS(2,3)*ELS(1,1)
      - ELS(2,2)*ELS(1,1)),( - ELS(2,4)*ELS(1,1))/(
     ELS(2,4)*ELS(1,3) - ELS(2,4)*ELS(1,2) + ELS(2,4)*ELS(1,1)
      + ELS(2,3)*ELS(1,1) - ELS(2,2)*ELS(1,1)),(ELS(1,1)
     *(ELS(2,3) - ELS(2,2)))/(ELS(2,4)*ELS(1,3)
      - ELS(2,4)*ELS(1,2) + ELS(2,4)*ELS(1,1)
      + ELS(2,3)*ELS(1,1) - ELS(2,2)*ELS(1,1))),
  (( - ELS(2,4)*ELS(1,2) - ELS(2,3)*ELS(1,2) + ELS(2,2)*ELS(1,3))
   /(ELS(2,4)*ELS(1,3) - ELS(2,4)*ELS(1,2) + ELS(2,4)*ELS(1,1)
     + ELS(2,3)*ELS(1,1) - ELS(2,2)*ELS(1,1)),(ELS(2,4)*ELS(1,3)
      + ELS(2,4)*ELS(1,1) + ELS(2,3)*ELS(1,1))/(
     ELS(2,4)*ELS(1,3) - ELS(2,4)*ELS(1,2) + ELS(2,4)*ELS(1,1)
      + ELS(2,3)*ELS(1,1) - ELS(2,2)*ELS(1,1)),(
      - (ELS(2,4)*ELS(1,2) + ELS(2,2)*ELS(1,1)))/(
```

```
    ELS(2,4)*ELS(1,3) - ELS(2,4)*ELS(1,2) + ELS(2,4)*ELS(1,1)
     + ELS(2,3)*ELS(1,1) - ELS(2,2)*ELS(1,1)),(
    ELS(2,3)*ELS(1,2) - ELS(2,2)*ELS(1,3) - ELS(2,2)*ELS(1,1))/
 (ELS(2,4)*ELS(1,3) - ELS(2,4)*ELS(1,2) + ELS(2,4)*ELS(1,1)
   + ELS(2,3)*ELS(1,1) - ELS(2,2)*ELS(1,1))),
(( - ELS(2,4)*ELS(1,3) - ELS(2,3)*ELS(1,2) + ELS(2,2)*ELS(1,3))
 /(ELS(2,4)*ELS(1,3) - ELS(2,4)*ELS(1,2) + ELS(2,4)*ELS(1,1)
    + ELS(2,3)*ELS(1,1) - ELS(2,2)*ELS(1,1)),(ELS(2,4)*ELS(1,3)
     + ELS(2,3)*ELS(1,1))/(ELS(2,4)*ELS(1,3)
     - ELS(2,4)*ELS(1,2) + ELS(2,4)*ELS(1,1)
     + ELS(2,3)*ELS(1,1) - ELS(2,2)*ELS(1,1)),(
     - ELS(2,4)*ELS(1,2) + ELS(2,4)*ELS(1,1)
     - ELS(2,2)*ELS(1,1))/(ELS(2,4)*ELS(1,3)
     - ELS(2,4)*ELS(1,2) + ELS(2,4)*ELS(1,1)
     + ELS(2,3)*ELS(1,1) - ELS(2,2)*ELS(1,1)),(
    ELS(2,3)*ELS(1,2) - ELS(2,3)*ELS(1,1) - ELS(2,2)*ELS(1,3))/
 (ELS(2,4)*ELS(1,3) - ELS(2,4)*ELS(1,2) + ELS(2,4)*ELS(1,1)
   + ELS(2,3)*ELS(1,1) - ELS(2,2)*ELS(1,1))),
((ELS(2,4)*(ELS(1,3) - ELS(1,2)))/(ELS(2,4)*ELS(1,3)
     - ELS(2,4)*ELS(1,2) + ELS(2,4)*ELS(1,1)
     + ELS(2,3)*ELS(1,1) - ELS(2,2)*ELS(1,1)),(ELS(2,4)
    *ELS(1,1))/(ELS(2,4)*ELS(1,3) - ELS(2,4)*ELS(1,2)
     + ELS(2,4)*ELS(1,1) + ELS(2,3)*ELS(1,1)
     - ELS(2,2)*ELS(1,1)),( - ELS(2,4)*ELS(1,1))/(
    ELS(2,4)*ELS(1,3) - ELS(2,4)*ELS(1,2) + ELS(2,4)*ELS(1,1)
     + ELS(2,3)*ELS(1,1) - ELS(2,2)*ELS(1,1)),(ELS(1,1)
    *(ELS(2,3) - ELS(2,2)))/(ELS(2,4)*ELS(1,3)
     - ELS(2,4)*ELS(1,2) + ELS(2,4)*ELS(1,1)
     + ELS(2,3)*ELS(1,1) - ELS(2,2)*ELS(1,1))))
```

The connectivity relationships between the flux control coefficients and the elasticity coefficients,

[0 0]
[0 0]
[0 0]
[0 0]

The summation relationships between the flux control coefficients. This will hold the relationship "C'* K = K",

[0 1]
[1 1]
[1 0]
[0 1]

The Matrix K is

[0 1]
[1 1]
[1 0]
[0 1]

},

{Concentration control coefficients,

```
MAT((( - ELS(2,4) - ELS(2,3) + ELS(2,2))/(ELS(2,4)*ELS(1,3)
   - ELS(2,4)*ELS(1,2) + ELS(2,4)*ELS(1,1)
   + ELS(2,3)*ELS(1,1) - ELS(2,2)*ELS(1,1)),ELS(2,4)/(
  ELS(2,4)*ELS(1,3) - ELS(2,4)*ELS(1,2) + ELS(2,4)*ELS(1,1)
```

```
 + ELS(2,3)*ELS(1,1) - ELS(2,2)*ELS(1,1)),( - ELS(2,4))/(
ELS(2,4)*ELS(1,3) - ELS(2,4)*ELS(1,2) + ELS(2,4)*ELS(1,1)
 + ELS(2,3)*ELS(1,1) - ELS(2,2)*ELS(1,1)),(ELS(2,3)
 - ELS(2,2))/(ELS(2,4)*ELS(1,3) - ELS(2,4)*ELS(1,2)
 + ELS(2,4)*ELS(1,1) + ELS(2,3)*ELS(1,1)
 - ELS(2,2)*ELS(1,1))),
((ELS(1,3) - ELS(1,2))/(ELS(2,4)*ELS(1,3) - ELS(2,4)*ELS(1,2)
 + ELS(2,4)*ELS(1,1) + ELS(2,3)*ELS(1,1)
 - ELS(2,2)*ELS(1,1)),ELS(1,1)/(ELS(2,4)*ELS(1,3)
 - ELS(2,4)*ELS(1,2) + ELS(2,4)*ELS(1,1)
 + ELS(2,3)*ELS(1,1) - ELS(2,2)*ELS(1,1)),( - ELS(1,1))/(
ELS(2,4)*ELS(1,3) - ELS(2,4)*ELS(1,2) + ELS(2,4)*ELS(1,1)
 + ELS(2,3)*ELS(1,1) - ELS(2,2)*ELS(1,1)),( - ELS(1,3)
 + ELS(1,2) - ELS(1,1))/(ELS(2,4)*ELS(1,3)
 - ELS(2,4)*ELS(1,2) + ELS(2,4)*ELS(1,1)
 + ELS(2,3)*ELS(1,1) - ELS(2,2)*ELS(1,1))))
```

The connectivity relationships between the concentration control coefficients and the elasticity coefficients

```
[-1  0 ]
[0   -1]
```

The summation relationships between the Concentration control coefficients

```
[0  0]
[0  0]
```

```
},
}}
```

Where

$$ELS(j,i) = \epsilon'_{ij} = \frac{\delta v_i}{\delta x_j} \tag{7.48}$$

Conclusions

A model has been presented for the application of computer algebra systems, such as REDUCE, to the solution of complex kinetic models of enzymes. It was demonstrated that it is possible to obtain some kinetic parameters for the steady state rate equation for complex enzyme systems from biochemical experimentation. However, it was also demonstrated that there are some important difficulties to be solve before the techniques become generally useful.

One of the major problem was the time required for the calculation of a compound rate law for a two enzyme system when the kinetic parameters were expressed symbolically rather than with numerical values (this is necessary for parameter estimation using the techniques described, though not for simulation). Although there were some difficulties, a solution of the coupled enzyme systems has been found using different method (resultant), but the problem was to generate the FORTRAN expression of the result printed out by REDUCE. On evaluation with specific values in floating point by the numerical analysis package (NAG) such expressions are prone to serious rounding, overflow and underflow errors. However, there are two way to evaluate such massive expressions into FORTRAN form; one of them is to use computer algebra system to evaluate the expressions analytically for the numerical analysis package as and when they are needed than to use a standard numerical analysis package for the numerical stage. The technique we have build up (see chapter 3) solve enzyme kinetic problems using a mixture of analytical and numerical techniques, but we could not apply the technique to all work which was handled in this thesis because the new technique takes nearly 10 times as long to run. Second approach was to reduce the size of the expression. To do this we improved a method to decide which kinetic parameters are important in the equation. The method was to look at the sensitivity of the overall rate law to variations in the kinetic parameters. Using the method we have chosen some parameters to fit the steady state rate law. However, the kinetic parameters obtained had a low standard deviation suggesting that they fitted the data well. This does not necessarily mean that they are "true values". On the other hand, this does not necessarily mean that something is wrong, since the values of the constants obtained are dependent on the

values of the other constants which were already numerically represented in the equation. the propagation of any errors in these would affect those symbolically represented—thus to obtain best fit estimates for any one constant, to a particular set of data, it may be advantageous to find simultaneously the best fit estimates for all the kinetic constants represented in the equation. An effective approach is to use a computer algebra system to perform to analytical stage and to use standard numerical analysis package for the numerical stage.

A more insidious problem was the selection of the NAG routines because many of them were usually failing to find the minimum of the function. We usually avoided using the NAG routines which need first and second derivatives of the given function to be evaluated, because expressions of the derivatives are so large. Evaluating these large expressions in FORTRAN is prone to rounding error. We used the NAG least square specialised routines $E04FDF$, $E04FCF$ and the Simplex ($E04FCF$) method to estimate kinetic parameters. These methods cannot be used unless the kinetic parameters are scaled. The kinetic parameters were scaled so that their order was unity.

The rate law governing the kinetics of a single enzyme mediated reaction may be derived relatively easily by hand given knowledge of the enzyme's mechanism. An alternative, relatively simple, schematic procedure, King-altman method, for deriving steady state rate laws. These mechanisms need not be very complex. Indeed a good example is the simple Michaelis-Menten mechanism which only contains two different enzymatic forms. An example program described in the chapter 7 provides a fast and easy to use way of obtaining rate-laws of arbitrarily complex enzymatic mechanisms. The ability of providing rate equations in several different representations is certainly of great use, especially when the objective is the experimental determination of the kinetic parameters. In this case, some sets of parameters are better for this purpose than others, but one cannot decide which a priori. Also, as relationships between the different representations are given, after determinating one set of parameters all other follow straightforward, even the rate constants of the elementary reaction steps. The rate laws are typically non-linear in the concentrations of metabolites involved. When

a number of enzymes interact the composite rate law for the complete system involves the simultaneous solution of the individual enzyme rate laws. We show how computer algebra can be used to solve this previously intractable problem, using the method of Gröbner Bases.

The method described to derive rate equations of enzyme catalysed reactions is mathematically accurate, fast and easy to use, given the availability of the REDUCE system, widespread in the academic world. It does not require any previous expertise with REDUCE. It is powerful as provides the user the choice of different (but equivalent) representations of the rate-laws.

Metabolic control analysis has been successfully used to understand how enzymes control fluxes and concentrations of intermediate metabolites. Metabolic control analysis has in some remarkable cases solved polemic discussions in biochemistry. Elasticity coefficients are a measure of the extent of how a reaction rate is affected by its effectors. In Reder's formalism [Reder, 1988] elasticities coefficients are just the partial derivatives and are therefore named *direct elasticity coefficients.* Advantage of the computer algebra techniques is the ability to derive the expression of the elasticity coefficients in terms of the kinetic parameters, like the rate equations are presented using different sets of parameters such that the user can choose the one which best suites his needs.

A computer program presented in appendix C, is more general in that it is based on the mathematical analysis of the control theory of metabolism. Using the program it is possible to calculate the control coefficients from the elasticity coefficients whatever the metabolic network. An other advantage of the our program is that it is now possible to determine all the summation and all the connectivity relationships between direct coefficients. It is clear that the relationships between direct coefficients can be translated into the corresponding relationships between the normalised ones. This would involve using ratios of rates, or ratios of concentrations over rates, and would therefore be more complex and not so easy to handle. This justifies the use of direct coefficients which is by no means a limitation. It is always possible to transform one type of coefficient into the other using the computer algebra system. Although Reder's program [Letellier

et al, 1991] is limited to a maximum of 19 steps and 19 intermediate metabolites, our program is now unlimited.

We have a complete methodology for estimating metabolic control parameters from simple time-course data, which works as follows:

1. Determine rate laws for the individual enzymes, either from the literature or from their published mechanism.

2. Solve individual rate laws simultaneously to obtain a composite rate law for the whole system.

3. Fit this rate law to time-course data to obtain estimates of the conventional kinetic parameters.

4. Using the definitions of metabolic control parameters compute formulae for them in terms of conventional kinetic parameters.

5. Substitute the estimates of conventional kinetic parameters from step 3 into the formulae for the metabolic control parameters to obtain their values.

We have improved a technique is completely general and can essentially be applied ta any problems which occur in phydics, engineering, biological seciences, chemistry, petroleum engineering and medical seciences.

The fundamental aim of this work is very simple. We believe that people with mathematical problems want mathematical solutions and, provided that they can be relied upon, are generally particularly interested in where they came from enzyme kinetic. We believe the work described here is a major step towards that goal.

Appendix A

An Example Program to Calculate Enzyme Kinetic Rate Law

A derivation of a rate law using Groebner bases. We write down equations for the rate of formation of all the intermediates, the rate of production of the product and the constraint that the amount of enzyme is fixed in terms of individual rate constants. We then eliminate all the intermediate and free enzyme concentrations. Finally we match against known forms of the initial and reverse reactions, to get a rate law in familiar terms.

```
We used here malate  dehydrogenase as an example.
%                      k1
%    E + NADH + oaa  <----->  E.NADH + oaa
%                      k2
%
%                      k3
%    E.NADH   + oaa  <----->  E.NADH.oaa
%                      k4
%
```

```
%                        k5
%    E.NADH.oaa       <----->  E.NAD + malate
%                        k6
%
%                        k7
%    E.NAD + malate   <----->  E + NAD + malate
%                        k8
% Load  Groebner basis package.
  load!-package 'groebner $
% Rate of production of malate is
  v!_malate := k5 * enz!_nadh!_oaa - k6 * enz!_nad * malate $
% Rates of formation of intermediates are
  v!_enz!_nadh       := k1 * enz * nadh -
                        k2 * enz!_nadh +
                        k4 * enz!_nadh!_oaa -
                        k3 * enz!_nadh * oaa $
  v!_enz!_nadh!_oaa := k3 * enz!_nadh * oaa -
                        k4 * enz!_nadh!_oaa +
                        k6 * enz!_nad * malate -
                        k5 * enz!_nadh!_oaa $
  v!_enz!_nad        := k5 * enz!_nadh!_oaa -
                        k6 * enz!_nad * malate +
                        k8 * enz * nad -
                        k7 * enz!_nad $
% Constraint on the enzyme total - the free enzyme + intermediates is
% a  constant, E0.
  constraint := enz0 - (enz + enz!_nadh + enz!_nadh!_oaa + enz!_nad) $
% Now eliminate all the enzyme species, to give a rate, v, which is
% equal to  the rate of formation of malate
  bas := groebner( { num( v - v!_malate ),
```

```
                    num( v!_enz!_nadh ),
                    num( v!_enz!_nadh!_oaa ),
                    num( v!_enz!_nad ),
                    num( constraint ) },
                  { enz, enz!_nadh, enz!_nadh!_oaa, enz!_nad, v } ) $
% Extract the rate law from the last element of bas
  veq  := num( part( bas, length( bas ))) $
  vsol := solve( veq, v ) $
% Rate law in terms of individual rate constants.
  v    := part( vsol, 1, 2 ) ;
% We now have a rate law in terms of individual rate constants. We now
% derive it in terms of Vmax's (in forward and reverse directions) and
% Michaelis  constants. For a general reaction
%                  A + B  <----->  C + D
% It is convenient to write the initial forward rate law as
%          1        1          K          K             K
%                                A          B             AB
%        ---  =  -----  +  -------  +  -------  +  -------------
%                  V       V . [A]     V . [B]     V . [A] . [B]
%          v        f       f           f           f
% We can write an equivalent rate law for the reverse direction. First
% of all  we obtain the inverses of the computed rate law.
  vfinv :=  1 / sub( malate = 0, nad = 0, v ) $
  vrinv := -1 / sub( oaa = 0, nadh = 0, v ) $
% Now the desired laws
  vfdinv := 1              /  vf +
            k!_oaa         / (vf * oaa) +
            k!_nadh        / (vf * nadh) +
            k!_oaa!_nadh   / (vf * oaa * nadh) ;
  vrdinv := 1              /  vr +
```

```
            k!_malate      / (vr * malate) +
            k!_nad         / (vr * nad) +
            k!_malate!_nad / (vr * malate * nad) ;
% Now vfinv must hold for all concentrations of oaa and nadh and vrinv
% for all concentrations of malate and nad, whether we express the
% rate law in terms of Vmax's and Km's or in terms of individual rate
% constants. Thus the coefficients of the metabolites must be
% identical however expressed. We thus write
%                          vfinv == vfdinv
% and equate powers of the metabolites on each side. For convenience
% we remove  denominators, so we in fact equate powers for
%       num(vfinv) * den(vfdinv) == num(vfdinv) * den(vfinv)
            eq1f := num(vfinv) * den(vfdinv) $
            eq2f := num(vfdinv) * den(vfinv) $
            eq1r := num(vrinv) * den(vrdinv) $
            eq2r := num(vrdinv) * den(vrinv) $
% All of these are quadratic in the coefficients, so equate first and
% second degree. This will give us 8 equations, which we can then
% solve for the eight  unknowns.
       s1  := coeffn(coeffn(eq1f,nadh,1),oaa,1) -
              coeffn(coeffn(eq2f,nadh,1),oaa,1) $
       s2  := coeffn(coeffn(eq1f,nadh,1),oaa,2) -
              coeffn(coeffn(eq2f,nadh,1),oaa,2) $
       s3  := coeffn(coeffn(eq1f,nadh,2),oaa,1) -
              coeffn(coeffn(eq2f,nadh,2),oaa,1) $
       s4  := coeffn(coeffn(eq1f,nadh,2),oaa,2) -
              coeffn(coeffn(eq2f,nadh,2),oaa,2) $
       s5  := coeffn(coeffn(eq1r,nad,1),malate,1) -
              coeffn(coeffn(eq2r,nad,1),malate,1) $
       s6  := coeffn(coeffn(eq1r,nad,1),malate,2) -
```

```
            coeffn(coeffn(eq2r,nad,1),malate,2) $
        s7  := coeffn(coeffn(eq1r,nad,2),malate,1) -
            coeffn(coeffn(eq2r,nad,2),malate,1) $
        s8  := coeffn(coeffn(eq1r,nad,2),malate,2) -
            coeffn(coeffn(eq2r,nad,2),malate,2) $
        equ := { s1, s2, s3, s4, s5, s6, s7, s8 }$
        var := { k1, k2, k3 ,k4, k5, k6, k7, k8 }$
        bas := groesolve(equ, var) $
        ll  := length( bas ) $
        bas1 := part( bas, ll) ;
        k1 := rhs( part( bas1, 1 ) ) $
        k2 := rhs( part( bas1, 2 ) ) $
        k3 := rhs( part( bas1, 3 ) ) $
        k4 := rhs( part( bas1, 4 ) ) $
        k5 := rhs( part( bas1, 5 ) ) $
        k6 := rhs( part( bas1, 6 ) ) $
        k7 := rhs( part( bas1, 7 ) ) $
        k8 := rhs( part( bas1, 8 ) ) $
% Lets substitute value of k1, ..., k8.
        vfinv - vfdinv ;
        vrinv - vrdinv ;
%  And the full rate law
        v := v ;
end $
```

Appendix B

A REDUCE Program to Calculate Conservation Relationships

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%    This program calculates conservation relationships of the
% given metabolic network. The  program is based on  Reder's methods
% [Reder, 1988].
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
   procedure findindep( n ) ;
% n  = Stoichiometry matrix of the metabolic network. Dimensions rxc.
% rn = number of independent metabolites in the network and rank of
%      the  matrix n.
% uz = a list of the number of rows and columns in the matrix n.
% r  = number of rows in the matrix n.
% c  = number of columns in the matrix n.
% cc = identity matrix.
% rr = matrix extracted from n with rn independent rows of n.
%      Dimensions rnxc.
```

```
%  r0 = is a square invertible matrix. Dimensions rnxrn.
%  r11 = matrix. Dimensions rnx(c-rn).
        begin ;
           nn := n           $    %  copy the matrix n to nn.
           rn := rank( nn )  $    %  rank of the matrix nn.
           uz := length( n ) $    %  dimension of the matrix n.
           r  := part(uz, 1) $    %  number of rows in the matrix n.
           c  := part(uz, 2) $    %  number of columns in the matrix n.
%  declaration of the matrixes.
           matrix cc(c, c), rr(rn, c)   $
           matrix r0(rn, rn), ns(r, rn) $
           if (c-rn) > 1
              then << matrix r11(rn, (c-rn)) >> $
           matrix nss(rn,r) $
           matrix px1(r,c) $
%  find the independent rows of the given stoichiometry matrix n.
           if ( r=rn )
              then << rr   := nn $
                      px21 := nn >>
              else << px1  := flir nn $
                      px19 := tp(px1) $
                      px20 := flir px19 $
                      px21 := tp(px20)$
                      << for i:= 1: rn  do
                           << for j := 1: c do
                                << rr(i,j) := px21(i,j) >>
                           >>
                      >> $
                   >> $
%  Now we find the matrix r0, then calculate the matrix r11. It is
```

```
%  calculated  using the ralationship
%                          RR = | R0.R11 |
          for i := 1: rn  do
             << for j := 1: rn  do
                 << r0(i,j) := rr(i,j) >>
             >> $
          if (c-rn) > 1
             then << for i := 1: rn do
                       << for j := 1: (c-rn) do
                             << r11(i,j) := rr(i,j+rn) >>
                        >>
                  >>
             else<<  r11 := 0 >> $
          if r neq rn
              then <<
%  Calculation of the matrix N*  which is exracted from N in which the
%  first  rn columns are independents.
                       px5 := tp(px21) $
                       px6 := flir px5 $
                       for i := 1: rn  do
                         << for j := 1: r  do
                              << nss(i,j) := px6(i,j) >>
                         >> $
                       ns := tp(nss) $
%  The matrix L0 which is determined with rn independent columns of
%  NN. i.e.  on the matrix N* and R0.
%                       L0 = N*.R0^(-1)
                       lo := ns * ( 1 / r0 ) $
                    >> $
          if ( r=rn )
```

```
            then << lo := nn $
                    rtu := ( {"Write matrix X considering matrix
                              N which is =", lo })$
                              return rtu >>
            else << rty := ( {"Write matrix X considering matrix
                              N which is =", ns }) $
                              return rty >>
      end; % end of procedure findindep.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
      procedure conser( x ) ;
        begin;
            matrix xr(rn,1), ll0((r-rn), rn), x0((r-rn), 1);
            xv := x ;
            for i :=1: rn  do
                << for j := 1: 1 do
                      << xr(i,j) := xv(i,j) >>
                >> $
            for i :=1: (r-rn)  do
                << for j := 1: 1 do
                      << x0(i,j) := xv((rn+i),j) >>
                 >> $
            for i :=1: (r-rn)  do
                 << for j := 1: rn  do
                      << ll0(i,j) := lo((rn+i),j) >>
                 >> $
% Conservation relationships for given system.
            result := x0-ll0*xr $
            return result
      end; % end  procedure  conser
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%'vector' here means a LISP vector with numeric entries in coords 1-upbv
 symbolic procedure FindLinIndepRows lbasis;
  begin scalar obasis, n, v, muij, w, d, thisrow, lo, hi, tmp;
    n := upbv lbasis;
    d := mkvect n;
    putv(d, 0, 1);
    obasis := mkvect n;
    lo := 1;
    hi := n;
    while lo <= hi do
    << v := getv(lbasis, lo);
       thisrow := copy v;
       for j := 1:lo-1 do
       %%% at the start of this loop v is d<j-1> times its true value.
       << w := getv(obasis, j); %this is d<j-1> times the true value.
          muij := dot(w, v)/getv(d, j-1); %this is d<j> times the true value
          v := scalarprod(getv(d, j), v);
          %%%v is now d<j>*d<j-1> times the true value
          if not zerop muij
            then v := vecdifference(v, scalarprod(muij, w));
          v := scalarquot(getv(d, j-1), v); %v is now d<j> times true value.
       >>;
       if dot(v, v) neq 0 then
       << putv(obasis, lo, v); %d<i-1> times the true value
          putv(d, lo, dot(v, v)/getv(d, lo-1));
          lo := lo+1
       >>
       else << putv(obasis, lo, getv(obasis, hi));
               tmp := getv(lbasis, lo);
               putv(lbasis, lo, getv(lbasis, hi));
```

```
            putv(lbasis, hi, tmp);
            hi := hi-1
          >>;
    >>;
    return (lo-1) . lbasis
  end;
 symbolic procedure vecdifference(a, b);
% a, b are vectors : result is their difference.
  begin scalar ans;
    ans := mkvect upbv a;
    for i := 1:upbv a do putv(ans, i, getv(a, i) - getv(b, i));
    return ans
  end;
 symbolic procedure dot(a, b);
% a, b are vectors : result is the inner prod of a & b as a rational.
  begin scalar ans;
    ans := 0;
    for i := 1: upbv a do ans := ans + getv(a, i)*getv(b, i);
    return ans
  end;
 symbolic procedure scalarprod(s, a);
% s is a number, a is a vector : result is s.a .
  begin scalar ans;
    ans := mkvect upbv a;
    for i := 1:upbv a do putv(ans, i, s*getv(a, i));
    return ans
  end;
 symbolic procedure scalarquot(s, a);
% s is a number, a is a vector : result is (1/s).a .
  begin scalar ans;
```

```
    ans := mkvect upbv a;
    for i := 1:upbv a do putv(ans, i, getv(a, i)/s);
    return ans
  end;
 symbolic procedure l2v l;
  begin scalar ans;
    if atom cdr l then return car l;
    ans := mkvect length l;
    for i:=1: upbv ans do
    << putv(ans, i, l2v car l);
       l := cdr l;
    >>;
    return ans;
  end;
 symbolic procedure v2l v;
  begin scalar ans;
    if not vectorp v then return v;
    for i := upbv v step -1 until 1 do
      if not null getv(v, i) then ans := v2l getv(v, i) . ans;
    return ans;
  end;
lisp operator flir;
 symbolic procedure flir x;
% x is a matrix of integers: (mat (a11 ... a1n) (a21 ...) ... (... ann))
% Result is prefix form (?) for lin indep row of this matrix.
% The rows are taken to be the vectors
  begin scalar tmp;
    if car x neq 'mat then REDERR("flir only works on matrices of integers");
    tmp := for each row in cdr x collect
              for each entry in row collect entry ./ 1;
```

```
    tmp := FindLinIndepRows l2v tmp;
    print list("Rank was ", car tmp);
    tmp := v2l cdr tmp;
    return 'mat . tmp
  end; % end of the flir program.
end;  % end of the program
```

Appendix C

A REDUCE Program for Metabolic Control Analysis

A REDUCE program to calculate control coefficient, concentration control coefficients, and summation and connectivity theorems in terms of elasticity coefficients. The program is based on metabolic control theory and used the Reder's methods [Reder, 1988; Letellier et al, 1991].

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
   procedure control( n ) ;
%  n  = Stoichiometry matrix of the metabolic network. Dimensions rxc.
%  rn = number of independent metabolites in the network and rank of
%       the  matrix n.
%  uz = a list of the number of rows and columns in the matrix n.
%  r  = number of rows in the matrix n.
%  c  = number of columns in the matrix n.
%  cc = identity matrix.
%  rr = matrix extracted from n with rn independent rows of n.
%  Dimensions  rnxc.
%  r0 = is a square invertible matrix. Dimensions rnxrn.
```

```
%  r11= matrix. Dimensions rnx(c-rn).
        begin ;
           nn := n           $       %  copy the matrix n to nn.
           rn := rank( nn )  $       %  rank of the matrix nn.
           uz := length( n ) $       %  dimension of the matrix n.
           r  := part(uz, 1) $       %  number of rows in the matrix n.
           c  := part(uz, 2) $
%  declaration of the matrixes.
           matrix cc(c, c), rr(rn, c)   $
           matrix r0(rn, rn), ns(r, rn) $
           matrix r11(rn, (c-rn)), bb((c-rn), (c-rn)) $
           matrix kk(c, (c-rn)) $
           matrix nss(rn,r) $
           matrix px1(r,c) $
%  operator for elasticity coefficients.
           operator els $
%  find the independent rows of the given stoichiometry matrix n.
           if ( r=rn )
           then <<rr   := nn $
                  m    := nn $
                  px21 := nn >>
           else <<px1  := flir nn $
                  px19 := tp(px1) $
                  px20 := flir px19 $
                  px21 := tp(px20)$
                  m    := px21 $
                   << for i:= 1: rn  do
                      << for j := 1: c do
                         << rr(i,j) := px21(i,j) >>
                      >>
```

```
                >> $
              >> $
% Calculated auxiliary matrix BB.
          for i := 1: (c-rn) do
            << for j := 1: (c-rn) do
                 << if  i=j then bb(i,j) :=1 >>
            >> $
% Find the matrix r0 of which rank is equal to rank of given matrix.
           if r = rn
              then << for i := 1: rn  do
                        << for j := 1: rn  do
                             << r0(i,j) := rr(i,j) >>
                        >>
                   >>
              else <<for i := 1: rn  do
                          << for j := 1: rn  do
                               << r0(i,j) := rr(i,j) >>
                          >>
                   >> $
% Calculate the matrix r11. It is calculated using the ralationship.
%                        RR = | R0.R11 |
            if r = rn
               then << for i := 1: rn do
                         << for j := 1: (c-rn) do
                              << r11(i,j) := rr(i,j+rn) >>
                         >>
                    >>
               else <<  for i := 1: rn do
                         << for j := 1: (c-rn) do
                           << r11(i,j) := px21(i,j+rn) >>
```

```
                        >>
                      >> $
%  Calculated auxiliary matrix AA. It is calculated using the
%  ralationship
%                       AA = R0^(-1)*(-R11)
                        aa := r0^(-1)*(-r11) $
%  Calculate the matrix KK  of which form
%                              |  AA   |
%                       KK  =  |  BB   |
                 for i := 1: c do
                    << for j := 1: (c-rn) do
                        << if i>rn
                        then << kk(i,j) := bb(i-rn,j) >>
                        else << kk(i,j) := aa(i,j) >>
                        >>
                     >> $
               if r neq 1
                  then <<
%  Calculation of the matrix N*  which is exracted from N in which the first
%  rn columns are independents.
                         px5 := tp(px21) $
                         px6 := flir px5 $
                         for i := 1: rn  do
                            << for j := 1: r  do
                               << nss(i,j) := px6(i,j) >>
                            >> $
                          ns := tp(nss) $
%  The matrix L0 which is determined with rn independent columns of
%  NN. i.e.  on the matrix N* and R0.
%                         L0 = N*.R0^(-1)
```

```
                    lo := ns * ( 1 / r0 ) $
                  >> $
            if ( r=rn )
               then << lo := px21 >> $
%  Determination of a matrix of the elasticity coefficients
            for i := 1: r  do
               << for j := 1: c  do
                  << if m(i,j) = 0
                     then  << m(i,j) := 0 >>
                     else  << m(i,j) := els(i,j) >>
                  >>
               >> $
            mm := tp(m) ;
%  Identity matrix CC.
            for i := 1: c  do
              << for j := 1: c do
                  << if i =j
                     then << cc(i,j) := 1.0 >>
                  >>
              >> $
             if r = rn
               then << contr := cc-mm*((px21*mm)^(-1))*px21 $
                       metco := -((px21*mm)^(-1))*px21 $
                       conrel := contr*mm $
                       conrelcon := metco*mm $
                    >>
               else <<
%  Control coefficients of the given system which are calculated using
%  the  ralationship.
%                  CONTR  = CC-MM.LO.(RR.MM.LO)^(-1).RR
```

```
                    contr := cc-mm*lo*((rr*mm*lo)^(-1))*rr $
%  Concentration control coefficients of the given system which are
%  calculated using the ralationship.
%                   METCO  = -LO.(RR.MM.LO)^(-1).RR
                    metco := -lo*((rr*mm*lo)^(-1))*rr $
%  The connectivity relationships between the flux control
%  coefficients and  the elasticity coefficients.
%                   CONREL  = CONTR.(MM.LO)
                    conrel := contr*(mm*lo)  $
%  The connectivity relationships between the concentration control
%  coefficients and the elasticity coefficients.
%                   CONRELCON  = METCO.(MM.LO)
                           conrelcon  := metco*(mm*lo)  $
                         >> $
%  The summation relationships between the flux control coefficients.
                    sumcontrol  := contr*kk $
%  The summation relationships between the concentration control
%  coefficients.
                    sumconcent  := metco*kk $
%  The relationship between stoichiometry matrix N and matrix KK.
%                   N.KK = 0
                    rel1   := px21 *kk $
%  The relationship between RR and KK.
%                   RR.KK = 0.
                    rel2   := rr*kk $
%  Results
     results := ({ {" Matrix N  was changed and New N is ", px21,
                    "Flux Control Coefficients in matrix form", contr,
                    "The connectivity relationships between the
                     flux  control coefficients and  the elasticity
```

```
                    coefficients ",  conrel,
                   "The summation relationships between the flux
                    control coefficients. This will hold the
                    relationship  C' * K = K ",  sumcontrol ,"Matrix
                    K is ", KK },
                  {"Concentration control coefficients", metco,
                   "The connectivity relationships between the
                    concentration control  coefficients and the
                    elasticity coefficients", conrelcon,
                   "The summation relationships between the
                    Concentration control coefficients",
                    sumconcent },
                  { rel1, rel2 } }) $
      return results
      end; % end control
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%'vector' here means a LISP vector with numeric entries in coords
% 1-upbv.
 symbolic procedure FindLinIndepRows lbasis;
  begin scalar obasis, n, v, muij, w, d, thisrow, lo, hi, tmp;
    n := upbv lbasis;
    d := mkvect n;
    putv(d, 0, 1);
    obasis := mkvect n;
    lo := 1;
    hi := n;
    while lo <= hi do
    << v := getv(lbasis, lo);
       thisrow := copy v;
       for j := 1:lo-1 do
```

```
      %%% at the start of this loop v is d<j-1> times its true value.
      << w := getv(obasis, j); %this is d<j-1> times the true value.
         muij := dot(w, v)/getv(d, j-1); %this is d<j> times the true value
         v := scalarprod(getv(d, j), v);
         %%%v is now d<j>*d<j-1> times the true value
         if not zerop muij
           then v := vecdifference(v, scalarprod(muij, w));
         v := scalarquot(getv(d, j-1), v); %v is now d<j> times true value.
      >>;
      if dot(v, v) neq 0 then
      << putv(obasis, lo, v); %d<i-1> times the true value
         putv(d, lo, dot(v, v)/getv(d, lo-1));
         lo := lo+1
      >>
      else << putv(obasis, lo, getv(obasis, hi));
              tmp := getv(lbasis, lo);
              putv(lbasis, lo, getv(lbasis, hi));
              putv(lbasis, hi, tmp);
              hi := hi-1
           >>;
    >>;
    return (lo-1) . lbasis
  end;
 symbolic procedure vecdifference(a, b);
% a, b are vectors : result is their difference.
  begin scalar ans;
    ans := mkvect upbv a;
    for i := 1:upbv a do putv(ans, i, getv(a, i) - getv(b, i));
    return ans
  end;
```

```
 symbolic procedure dot(a, b);
% a, b are vectors : result is the inner prod of a & b as a
%  rational.
  begin scalar ans;
    ans := 0;
    for i := 1: upbv a do ans := ans + getv(a, i)*getv(b, i);
    return ans
  end;
 symbolic procedure scalarprod(s, a);
% s is a number, a is a vector : result is s.a .
  begin scalar ans;
    ans := mkvect upbv a;
    for i := 1:upbv a do putv(ans, i, s*getv(a, i));
    return ans
  end;
 symbolic procedure scalarquot(s, a);
% s is a number, a is a vector : result is (1/s).a .
  begin scalar ans;
    ans := mkvect upbv a;
    for i := 1:upbv a do putv(ans, i, getv(a, i)/s);
    return ans
  end;
 symbolic procedure l2v l;
  begin scalar ans;
    if atom cdr l then return car l;
    ans := mkvect length l;
    for i:=1: upbv ans do
    << putv(ans, i, l2v car l);
       l := cdr l;
    >>;
```

```
    return ans;
  end;
 symbolic procedure v2l v;
  begin scalar ans;
    if not vectorp v then return v;
    for i := upbv v step -1 until 1 do
      if not null getv(v, i) then ans := v2l getv(v, i) . ans;
    return ans;
  end;
lisp operator flir;
 symbolic procedure flir x;
% x is a matrix of integers: (mat (a11... a1n) (a21 ...) ... (... ann))
% Result is prefix form (?) for lin indep row of this matrix.
% The rows are taken to be the vectors
  begin scalar tmp;
    if car x neq 'mat then REDERR("flir only works on matrices of ntegers");
    tmp := for each row in cdr x collect
              for each entry in row collect entry ./ 1;
    tmp := FindLinIndepRows l2v tmp;
    print list("Rank was ", car tmp);
    tmp := v2l cdr tmp;
    return 'mat . tmp
  end; % end of the flir program.
end;  % end of the program
```

Bibliography

Acerenza, L, Sauro, H M and Kacser, H, (1989). Control Analysis of Time-dependent Metabolic System, *J. Theor. Biol.*, **137**, 423–444.

Achs, M J and Garfinkel, D, (1977). Computer Simulation of Energy Metabolism in Anoxic Perfused Rat Heart, *Am. J. Physiol.*, **232**, R164-R175.

Achs, M J and Garfinkel, D, (1982). Computer Simulation of Energy Metabolism in acidotic Cardiac Ischemia, *Am. J. Physiol.*, **242**, R533–R544.

Albe, K R, Butler, M H and Wright, B E, (1990). Cellular Concentrations of Enzymes and Their Substrates, *J. Theor. Biol.*, **143**, 163–195.

Alberty, R A, (1953). The Relationship between Michaelis Constants, Maximum Velocities and the Equilibrium Constant for an Enzyme-catalyzed Reaction, *J.Am.Chem.Soc.*, **75**, 1928–1932

Albiser, A M, Yamesaki, H, Brockheuse, H and Tiran, J, (1980). Hypercomplex Models of Insulin and Glucose Dynamics-Do They Predict Experimental Results? *Ann. Biomed. Eng.*, **8**, 538–577.

Askelof, D, Korsfeldt, M and Mannervick, B, (1976). Error structure of Enzyme Kinetic Experiments, *Eur. J. Biochem.*, **69**, 61–67.

Atkins, G L and Nimmo, I A, (1973). The Reliability of Michaelis constants and Maximum velocities Estimated by Using the Integrated Michaelis-Menten Equation, *Biochem. J.*, **135**, 779–784.

Atkins, G L, (1973). A simple Digital Computer Program for Estimating the Parameters of the Hill Equation, *Eur. J. Biochem.*, **33**, 175–180.

Bard, Y, (1974). Non-linear Parameter Estimation, *Academic Press, New York.*

Barnsley, E A, (1990). Steady State Kinetics of Enzymes with Substrates Having Multiple Reaction Sites, *J.Theor. Biol.*, **142**, 275-279

Bardsley, W G, (1977). The 3:3 Function in Enzyme Kinetics Possible Shopes of v/S and (1/v)/(1/S) Plots for Third Degree Steady-State Rate Equations, *J. Theor. Biol.*, **65**, 281-316.

Bardsley, W G and Wright, A J, (1983), A New Approach to the Measurement of Sigmoid Curves with Enzyme Kinetic and Ligand Binding Data, *J. Mol. Biol.*, **165**, 163-182.

Bates, D J and Frieden, C, (1973a). Small Computer Systems for Routine Analysis of Enzyme Kinetic Mechanisms. *Comput. Biomed. Res.*, **6**, 474–486.

Bates, D J and Frieden, C, (1973b). Treatment of Enzyme Kinetic Data III. Use of Full Time Course of a Reaction, as Examined by Computer Simulation, in Defining Enzyme Mechanisms, *J.Biol.Chem.*, **248**, 7878–7884.

Bates, D J and Frieden, C, (1973c). Full Time Course Studies on Oxidation of Reduced Coenzyme by Porcine Liver Glutamate DH: Use of Computer Simulation to Obtain

Rate and Dissociation Constants, *J. Biol. Chem.*, **248**, 7885–7890.

Bayram, M, Bennett, J P and Dewar, M C, (1991). Using computer algebra to determine rate constants in biochemistry, *11th conference of the School of Theoretical Biology, Limoges, France, 3–5 June, 1991.*

Bayram, M and Bennett, J P, (1992). A simple general purpose technique for interfacing between computer algebra and numerical analysis systems. Design and Implementation of Symbolic Computation Systems (DISCO-92), Bath, April 13–15, 1992.

Bennett, J P, Davenport, J H and Sauro, H M, (1988), Solution of Some Equations in Biochemistry, *Bath Computer Science Technical Report*, 88-12.

Bennett, J P and Fisher, D L, (1990). A Comparison of Two Methods of Analysing Enzyme Kinetics Using Computer Algebra, submitted to the International Symposium on Symbolic and Algebraic Computation, Hirosawa, Japan, August 1990

Bergmeyer, H U, (1981). Methods of Enzymatic Analysis, *Verlag Chemica*, vol, 1–2.

Bloomfield, V, Peller, L and Alberty, R A, (1962) Multiple Intermediates in Steadi-State Enzyme Kinetics. II. Systems Involving two Reactants and two products, *J. Am. Chem. Soc.*, **84**, 4367–4375.

Bock, H G (1981). Numerical treatment of inverse problems in chemical reaction kinetics. In *Modelling of chemical reaction systems*, ed. K H Ebert, P Deuflhard and W Jäger, Springer-Verlag, 1981, 102–125.

Boeker, E A, (1987) Analytical Methods for Fitting Integrated Rate Equations, *Biochem. J.*, **245**, 67–74.

Buchberger, B, (1985). A suvey on the method of Gröbner bases for solving problems in connection with systems of multi-variate polynomials. *Proceedings of the 2nd RIKEN Symposium on Symbolic and Algebraic Computation*, ed. N Inada and T Soma, World Scientific Publishing, 1985, 69-83.

Cascante, M, Caela, E I and Franco, R, (1990). Control Analysis of Systems Having two Steps Catalysed by the Same Protein Molecule in Unbranched Chains, *Eur. J. Biochem.*, **192**, 369–371.

Cha, S, (1968). A simple Method for Derivation of Rate Equations for Enzyme-catalyzed Reactions under the Rapid Equilibrium Assumption or Combined Assumptions of Equilibrium and Steady State, *J. Biol. Chem.*, **243**, 820–825.

Chance, B and Williams, G R, (1956). The Respiratory Chain and Oxidative Phosphorylation, *Adv. Enzymol.*, **17**, 65–134.

Chandler, J P, (1972). A program for Efficient Integration of Rate Equations and Least-squares Fitting of chemical Reaction Data, *Computers and Biomedical Research*, **5**, 515–534.

Cleland, W W, (1963). The Kinetics of Enzyme-catalysed Reactions with two or more Substrates or Products, *Biochimica et Biophysica Acta* , **67**, 104–137.

Cleland, W W, (1967). The Statistical Analysis of Enzyme Kinetic Data, *Adv. Enzymol.*, 1–32.

Cleland, W W, (1969). Statistical Analysis of Enzyme Kinetic Data, *Methods in Enzymol.*,**63**, 103–138.

Cornish-Bowden, A and Eisenthal, R, (1974). Statistical Considerations in the

Estimation of Enzyme kinetic Parameters by the direct Linear plot and Other Methods, *Biochem. J.*, **139**, 721–730.

Cornish-Bowden, A, (1977). An Automatic Method for Deriving Steady-State Rate Equations, *Biochem. J.*, **165**, 55-59

Cornish-Bowden, A and Wong, J T, (1978). Evaluation of Rate Constants for Enzyme-Catalysed Reactions by the Jackknife Technique, *Biochem. J.*, **175**, 969–976.

Cornish-Bowden, A, (1979). Fundamentals of Enzyme Kinetics, *Butterworth and Co (Publishers) Ltd. London.*

Cornish-Bowden, A and Endrenyi, L, (1981). Fitting Enzyme Kinetic Data without Prior Knowledge of Weights, *Biochem. J.*, **193**, 1005–1008.

Cornish-Bowden, A and Endrenyi, L, (1986). Robust Regression of Enzyme Kinetic Data.*Biochem. J.*, **234**, 21–29.

Cornish-Bowden, A and Wharton, C W, (1988). Enzyme kinetics. *Irl press, London.*

Crabbe, J, (1985). Computers in Biochemical Analysis, *Meth. Biochem. Anal.*, **31**, 417–474.

Crabbe, J, (1988). Simple Graphical Methods for Use with Complex Ligand-Binding and Enzyme Mechanisms. *FEBS lett.*, **235**, 183–188.

Crabbe, J, (1990). Correct Use of Scatchard Plots, *Trends Biochem.Sci.***15**, 12–13

Crabtree, B and Newsholme, E A, (1987). A systematic Approach to Describibg and Analysing Metabolic Control Systems, *Trends Biochem. Sci.*, **12**, 4–14.

Dalziel, K, (1957). Initial Steady State Velocities in the Evaluation of Enzyme-Coenzyme-Substrate Reaction Mechanisms, *Acta Chem. Scan.*, **11**, 1706–1723.

Davenport, J H, Siret, Y and Tournier, E, (1988). Computer Algebra. Systems and Algorithms for Algebraic Computation, *Academic Press*, London.

Delgado, J and Liao, J C, (1992). Metabolic Control Analysis Using Transient Metabolite Concentrations, *Biochem. J.*, **285**, 965–972.

Dewar, M C, (1991). Interfacing Algebraic and Numerical Computation, Ph.D Thesis. *University of Bath.*

Duggleby, R G and Morrison, J F, (1977). The Analysis of Progress Curves for Enzyme-Catalysed Reactions by Non-linear Regression, *Biochimica et Biophysia Acta*, **481**, 297–312.

Duggleby, R G, and Morrison, J F, (1978). Progress Curve Analysis in Enzyme kinetics,*Biochimica et Biophysia Acta*, **526**, 398–409.

Duggleby, R G, (1985). Estimation of the Initial velocity of Enzyme-Catalysed Reactions by Non-linear Regression Analysis of Progress Curves, *Biochem. J.*, **228**, 55–60.

Duggleby, R G, (1991). Analysis of Biochemical Data by Nonlinera Regression–Is It A Waste of Time, *Trends in Biochem. Sci.*, **16/2**, 51–52.

Edward, W Ng, (1979). Symbolic-Numeric interface: A review. *In Proceedings of EUROSAM'79, Springer-Verlag.* 330–345.

Efron, B, (1979). Bootstrap methods: Another look at the jackknife, *The Annal. of*

Statis., **7**, pp1-26

Eisenthal, R and Cornish-Bowden, A, (1974). The Direct Linear Plot: A new Graphical Procedure for estimating Enzyme Kinetic Parameters, *Biochem. J.*, **139**, 715–720.

Engel, P C, (1981). Enzyme Kinetics, *Chappon and Hall Ld.*, London and New York

Engel, P.C. and Ferdinand, W. (1973) The Significance of Abrupt Transitions in Lineweaver-Burk Plots with Particular Reference to Glutamate Dehydrogenase. *Biochem.J.* **131**, 97–105.

Fasella, P and Hammes, G G, (1967). A Temperature Study of Aspartate Aminotransferase, A reinvestigation, *Biochemistry*, **6**, 1798–1804.

Fell, D A and Sauro, H M, (1985). Metabolic Control and its Analysis: Additional Relationships Between Elasticities and Control Coefficients, *Eur. J. Biochem.*, **148**, 555-561.

Fell, D A and Sauro, H M, (1990). Metabolic control analysis. The effect of high enzyme concentrations, *Eur.J.Biochem.*, **37**, 1-5.

Fell, D A, (1992). Metabolic control analysis: a survey of its theoretical and experimental development, *Biochem. J.*, **286**, 313–330.

Fernley, H N, (1974). Statistical estimations in Enzyme Kinetics: The Integrated Michaelis Equation, *Eur. J. Biochem.*, **43**, 377–378.

Fisher, D L, (1990a). Applications of computer algebra to enzyme analysis, Technical Report 90-32, University of Bath, February 1990.

Fisher, D L, (1990b). Novel computer techniques in enzyme kinetics, Technical Report 90-41, University of Bath, August 1990.

Fisher, D D and Schultz, A R, (1969). Connection Matrix Representation of Enzyme Reaction Sequences, *Math. Biosci.* **4**, 189–200.

Fitch, J P, (1979). The Application of Symbolic Algebra to Physics–A Case of Creeping Flow, *In Proceedings of EUROSAM'79, Springer-Verlag*, 30–41.

Fitch, J P, (1985). Solving Algebraic Problems with REDUCE, *J. Symbolic Computation*, **1**, 211–217.

Fitch, J P, (1990). The Algebraic-Numeric Interface, *Computer Physics Communications,* **61**, 22–33.

Garfinkel, D, (1963). Digital Computer Simulation of Systems apparently Compartmented at the Cellular Level, *Ann. N. Y. Acad. Sci.*, **108**, 293–304.

Garfinkel, D, (1984). Modelling of Inherently Complex Biological Systmes: Problems, Strategies, and Methods, *Math.Biosci.*, **72**, 131–139.

Garfinkel, D, Garfinkel, L, Pring, M, Green, S B and Chance, B, (1970). Computer Applications to Biochemical Kinetics, *Ann. Rev. Biochem.*, **39**, 473-498.

Garfinkel, L, Kohn, M C and Garfinkel, D, (1979). Computer Simulation of the Fructose Bisphosphatasel Phosphofructokinase Couple in Rat Liver, *Eur. J. Biochem.*, **96**, 183–192.

Garfinkel, D, Marbach, C B and Shapiro, N Z, (1977). Stiff Differential Equations, *Ann. Rev. Biophys. Bioeng.*, **6**, 525–542.

Gates, B L, (1987). *GENTRAN user's manual.* The Rand Corportation, Santa Monica, California.

Gill, P E and Murray, W, (1978). Algorithms for the solution of the nonlinear least-squares problem, *SIAM J. Numer. Anal.*, **15**, 977-992.

Gill, P E, Murray, W and Wright, M H, (1981). Practical Optimization. *Academic press, London.*

Groen, A K, Wodars, R J A, Westerhoff, H V, Van der meer, R and Tager, J M, (1982). Quantification of the Contribution of Various Steps to the Control of Mitochondrial Respiration, *J.Biol. Chen.*, **257**, 2754–2757.

Haarhoft, K N, (1969). Use of Multivariate Non-linear Regression Analysis in Fitting Enzyme Kinetic Models. An Empirical Study of the Inhibition of Aspartate Transaminase by Dicarboxylic Acid substrate Analogues, *J. Theor. Biol.*, **22**, 117–150.

Hearn, A C, (1987). *Reduce User's Manual Version 3.* The Rand Corporation, Santa Monica, CA 90406.

Heinrich, R and Rapoport, T A, (1974). A Linear Steady State Treatment of Enzymatic Chains. General Properties, Control and Effector Strength, *Eur.J.Biochem.*, **42**, 89–95.

Heinrich, R, Rapoport, S M, (1983). The utility of mathematical models for the understanding of metabolic systems, *Biochem. Soc. Trans.*, **11**, 31-35.

Heinrich, R, Rapoport, S M, and Rapoport, T A, (1977). Metabolic Regulation and Mathematical Models, *Prog. Biophys. Mol. Biol.*, **32**, 1–82.

Hofmeyr, J S, Kacser, H and Merwe, J, (1986). Metabolic Control analysis of Moiety-Conserved cycles, *Eur. J. Biochem.*, **155**, 631-641.

Innis, G and Rexslad, E, (1983). Simulation Model Simplification Techniques, *Simulation*, **41**, 7-15.

Jenkins, W T and Fonda, M L, (1984). In *Transaminases*, Eds. Christen,P and Metzler, D Ei, pp 216–227, John Wiley and Sons, New York.

Kacser, H, (1989). Quantitative Variation and the Control Analysis of Enzyme Systems, *Evolution and Animal Breeding ,C.A.B, international, Oxfort, W.G.Hill and T.F. Mackay*, 219-226.

Kacser, H and Burns, J A, (1973). The Control of Flux, *Symp.Soc.Exp.Biol.*,**27**, 65–104.

Kacser, H and Burns, J A, (1979). Molecular Democracy: Who Shares the Controls? *Biochem. Soc. Trans.*, **7**, 1149-1160.

Kacser, H and Porteous, J W, (1987). Control of Metabolism: What Do We Have to Measure? *Trends Biochem. Sci.*, **12**, 5–14.

Kacser, H, (1983). The control of enzyme systems in *vivo*: Elasticity analysis of the steady state, *Biochem. Soc. Trans.*, **11**, 35-43.

Kacser, H, Sauro, H M and Acerenza, L, (1990). Enzyme-Enzyme interactions and Control Analysis, *Eur. J. Biochem.*, **187**, 481–491.

Kalkbrener, M, (1987). Solving Systems of Algebraic Equations by using Gröbner Bases, *EUROCAL'87*, Ed. J H Davenport, Leipzig, GDR, june 2-5, 282–292.

Kholodenko, B N, (1988). How do external parameters control fluxes and concentrations of metabolites? *FEBS*, **232/2**, 383-386

King, E L and Altman, C, (1956). A schematic method of deriving the rate laws for enzyme-catalyzed reactions, *J. Phs. Chem.*, **60**, 1375-1381.

Kistiakowsky, G B and Shaw, W H R, (1953). On the Mechanism of the Inhibition of Urease, *J. Am. Chem. Soc.*, **75**, 866–871.

Lam, C F and Priest, D G, (1972). Enzyme kinetics: Systematic Generation of Valid King-Altman Patterns, *Biophys. J.* **12**, 248–256.

Lam, C F and Schatz, N, (1978). Automatic Generation of Steady-State Enzyme Rate Equation and the Associated Constrant Equations, *J. Biochem.*, **84**, 585-595.

Leatherbarrow, R J, (1990). Using Linear and Non-linear Regression to Fit Biochemical Data, *Trens in Biochem. Sciences*, **15**, 455–459.

Letellier, T, Reder, C and Mazat, J P, (1991). Control–Software for the Analysis of the Control of Metabolic Networks, *Comput. Appl. Biosci.*, **49**, 8-1–810.

Malpiece,Y and Sharan, M, (1980). Theoretical and Experimental Study of an Immobilized Bienzyme system (computer calculations and electron microscopy visualization of local concentration profiles), *The J. of Biolog. Chem.*, **255**, 6883-6890.

Melendez-Hevia, E, Torres, N V, Sicilia, J and Kacser, H, (1990). Control analysis of transition times in metabolic systems, *Biochem. J.*, **265**, 195-202.

NAG, (1988). *NAG FORTRAN Library Manual - Mark 13, vols I-VI* Numerical Algorithms Group, Oxford.

NAG Ltd. (1990) The NAG Fortran Library Manual-Mark 14.

Nelder, J A and Mead, R, (1965). A simplex Method for Function Minimization, *Comput.*, **3-7**, 308–313.

Parkinson, J M and Hutchinson, D, (1972). An Investigation into the Efficiency of Variants on the Simplex Method, *In: Numerical Methods for Nonlinear Optimisation, Lootsma, FA(ed.)*, Academic Press, 1972, 115–135.

Raksanyi, A, Lecourtier, Y and Walter, E, (1986) Identifiability and Distinguishability Testing Via Computer Algebra. *Mathe. Bios.*, **77**, 245–266.

Rapoport, T A, Heinrich, R and Rapoport, S M, (1976). The Regulatory principles of Glycolysis in Eryhrocytes *in vivo* and *in vitro*. A minimal Comprehensive Model describing Steady states, Quasi Steady States and time-dependent Processses. *Biochem. J.*, **154**, 449–469.

Raval, D and Wolfe, R G, (1962). Malic Dehydrogenase. III. Kinetic Studies of the Reaction Mechanism by Product Inhibition. *Biochemistry* , **1**, 1112–1117.

Reder, C, (1988). Metabolic Control Theory: A Structural Approach, *J.Theret. Biol.*, **135**, 175–201.

Reich, J G, (1970). Parameter Estimation and Enzyme Kinetic Models, *FEBS Letters*, **9**, 245–251.

Reich, J G, Wangermann, G, Falck, M and Rohde, K, (1972). A General Strategy for Parameter Estimation from Isosteric and Allosteric Kinetic Data and Binding Measurements, *Eur. J. Biochem.*, **26**, 368–379.

Sauro, H M, Small, J R and Fell, D A, (1987). Metabolic Control and its Analysis: Extensions to the Theory and Matrix Method. *Eur.J.Biochem.*, **165**, 215–221.

Savageau, M A, (1971). Parameter sensitivity as a Criterion for Evaluating and Comparing the Performance of Biochemical systems, *Nature*, **229**, 542–544.

Segel, I H, (1975). Enzyme Kinetics. *A Wiley interscience Publication*, London.

Simmons, G F, (1963). Topology and Modern Analysis, *McGraw-Hill Book Company*, London.

Small, J R and Fell, D A, (1989). The Matrix Method of Metabolic Control Analysis: its Validity for Complex Pathway Structures, *J.Theor. Biol.*, **136**, 181-197.

Small, J R and Fell, D A, (1990). Metabolic Control Analysis: Sensitivity of Control coefficients to elasticities, *Eur. J. Biochem.*, **191**, 413–420.

Stoer, J, (1971). On the numerical solution of constrained least-squares problems. *SIAM J. Numer. Anal.*, **8/2**, 382-411.

Storer, A C and Cornish Bowden, A, (1974). The Kinetics of Coupled Enzyme Reactions, *Biochem. J.*, **141**, 205-209.

Storer, A C, Darlison, M G and Cornish-Bowden, A, (1975). The Nature of Experimental Error in Enzyme Kinetic Measurements, *Biochem. J.* , **151**, 361–367.

Swarz, J and Bremermann, H, (1975). Discussion of Parameter estimation in Biological Modelling, *J. Math. Biol.*, **1**, 241–257.

Taylor, J R, (1982). An Introduction to Error Analysis, *University science books, Mill Valley, California.*

Varon, R, Havsteen, B H, Gorcia, M, Valero, E and Canovas, F G, (1990). Derivation of the transient phase equations of enzyme mechanisms from those of other systems, *J. theor. Biol.*, **143**,pp 251-268.

Velick, S F and Vavra, J, (1962). A Kinetic and Equilibrium Analysis of the Glutamic Oxaloacetate Transaminase Mechanism, *J. Biological Chem.*, **237**, 2109–2122.

Waight, R D, Leff, P and Bardsley, W G, (1977). Steady State Kinetic studies of the Negative Cooperativity and Flip-Flop Mechanism for E.coli Alkaline Phosphatase, *Biochem. J.*, **167**, 787–798.

Waley, S G, (1981). An easy Method for the Determination of Initial Rates, *Biochem. J.*, **193**, 1009–1012.

Waser, M R, Garfinkel, L, Kohn, M C and Garfinkel, D, (1983). Computer modelling of Muscle Phosphofructokinase Kinetics, *J. Theor. Biol.*, **103**, 295–312.

Westerhoft, H V and Chen, Y, (1984). How do Enzyme Activities Control Metabolite Concentrations? *Eur. J. Biochem.*, **142**, 425-430.

Wilkinson, G N, (1961). Statistical Estimations in Enzyme Kinetics, *Biochem. J.*, **80**, 324–332.

Wong, J T F and Hanes, C S, (1962). Kinetic Formulations for Enzymic reactions Involving two Substrates, *Can. J. Biochem. Physiol.*, **40**, 763–803.